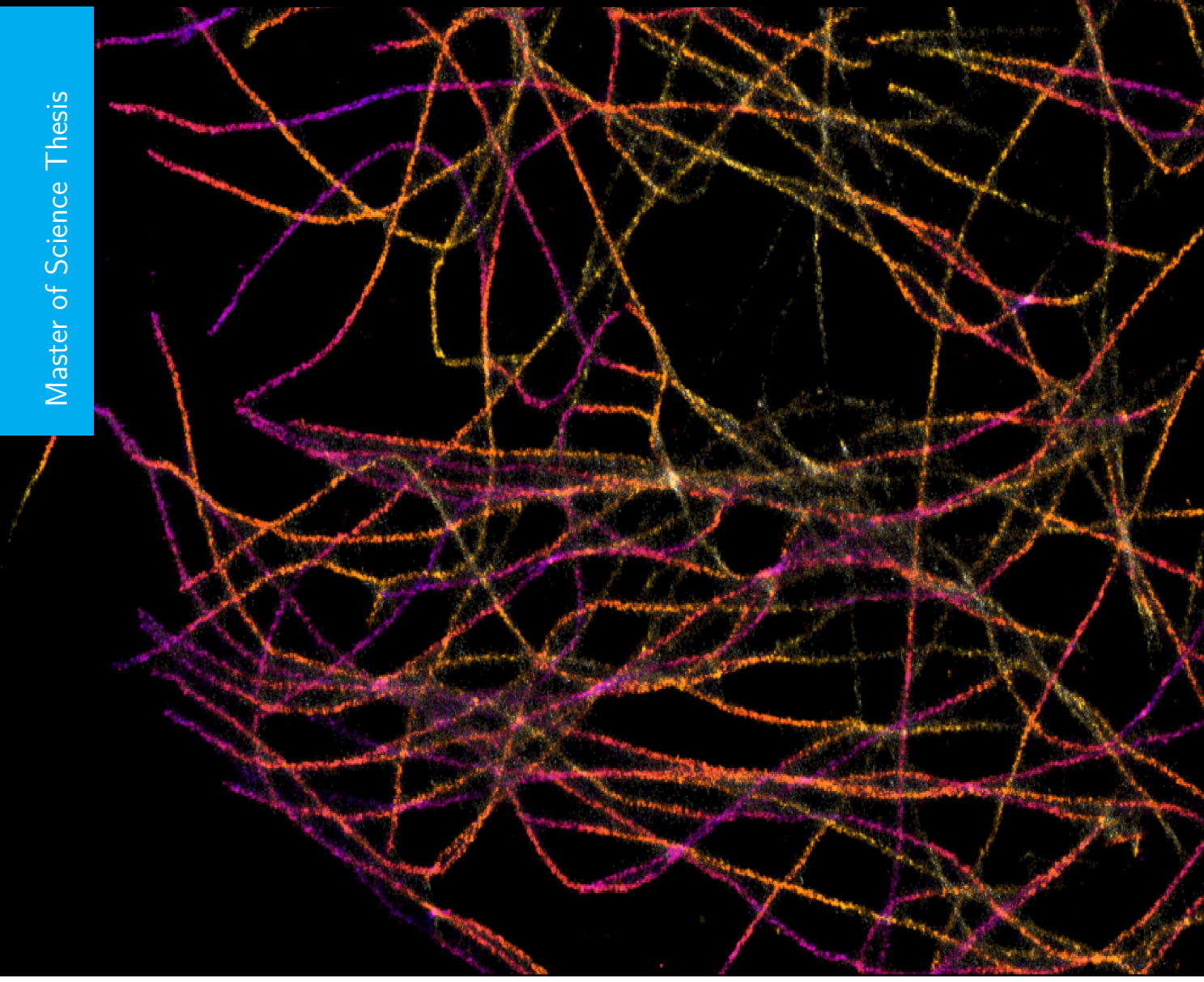
Data-Driven PSF Modeling using B-Splines

Willem M. Melching

Master of Science Thesis



TU Delft Delft University of Technology

Delft Center for Systems and Control

Data-Driven PSF Modeling using B-Splines

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Willem M. Melching

April 8, 2019

Faculty of Mechanical, Maritime and Materials Engineering (3mE)
Delft University of Technology

Image on the front page from http://jos3.umif.de/index.php/downloads/gallery/olympus-smlm

Abstract

Super Resolution Microscopy is a technique to image objects with resolution higher than the diffraction limit, the fundamental limit to the resolution of a microscope. One of the techniques used for Super Resolution Microscopy is Single Molecule Localization Microscopy (SMLM). For this method, a sample is labeled using fluorophores (fluorescent proteins or dyes), and each individual fluorophore is imaged individually. The fluorophores can be localized with a precision much higher than the diffraction limit, since the localization precision is only limited by the amount of photons collected. By combining the localizations of all the fluorophores, the complete object can be reconstructed with resolution higher than the diffraction limit [1].

To achieve a high localization precision, a model for the Point Spread Function (PSF) of the microscope is required. The PSF describes how the microscope distorts the image of the fluorophore. This model is used to estimate the fluorophore location in the sample. This means the model needs to be very accurate and unbiased. Furthermore, because an experiment contains hundred of thousands of images and the model needs to be evaluated multiple times per localization, the model also needs to be fast to evaluate.

Traditional fitters model the PSF with a Gaussian approximation. However, more recent studies showed the errors introduced by this approximation [2]. More accurate models have been introduced, such as C-Spline based approximations [3, 4]. These C-Splines require over a million spline coefficients, and therefore lots of calibration data. In this research, it is investigated if Simplex B-Splines can be used to better approximate the PSF. Simplex B-Splines consist of multiple B-Splines defined on a triangulation of simplices [5]. This allows for a very flexible model structure and high accuracy using a relatively low number of parameters.

Most applications assume the PSF is constant over the field of view. However, due to varying aberrations or refractive index mismatches the PSF actually changes with the fluorophore position in the field of view [6, 7, 8]. If these changes in the PSF are not corrected for, the precision is reduced or a significant bias is introduced. B-splines allow the model to be extended to higher dimensions, and can therefore also model these changes to the PSF.

Different metrics are used to compare models. Using a statistical (χ^2) test the model quality can be assessed. Localization precision is verified by implementing the model in a Maximum Likelihood Estimation (MLE) algorithm, and by showing that the standard deviation reaches the theoretical lower bound for precision, the Cramér-Rao Lower Bound (CRLB).

In this research it is shown that the performance of B-Spline models is comparable to state-of-the-art methods, but uses 144 times fewer spline coefficients. Furthermore, the B-Splines were also successfully extended to higher dimensions, to reduce the effect of a varying PSF.

Contents

List of Figures

Preface

Before I started my thesis I thought it would be much like my previous courses, but with a larger report at the end. I couldn't have been more wrong. Working a year on your thesis teaches you a lot of new skills, such as quickly understanding a scientific paper and deeper understanding of the theory behind your research.

However, the most valuable thing I learnt during my research was a proper research methodology. If you have a hypothesis, it is not enough to quickly write a simulation that proves your hypothesis and call it a day. You need to come up with an experiment, and predict what the results should look like. You also need to come up with a statistical test to prove your results were not produced by chance.

I also learnt that experimental data is always slightly different from the simulated data, and this will always mess up your results. Even if you think you have accounted for all physical processes in your simulations, there is always something that throws off your very well thought out algorithms.

All of this research would not have been possible without my supervisor, Carlas Smith. His door was always open for questions, and I could always walk in for a cup of coffee and some advice. He was always interested in my latest results, or problems I was having. This really helped my research get to a higher level.

I also would like to thank Coen de Visser. His research on B-Splines formed the basis of my research. He also wrote a great MATLAB toolbox to fit and evaluate B-Splines. My daily supervisor, Jelmer Cnossen, was also of great help. He wrote a localization microscopy toolbox, which allowed easy implementations of new PSF models in CUDA. We often had coffee together, talked about our research or why Python is a much better programming language than MATLAB.

Delft, University of Technology
April 8, 2019

Willem M. Melching

“ In theory, theory and practice are the same. In practice, they are not.”
— *Jan L.A. van de Snepscheut*

Chapter 1

Introduction

The purpose of this thesis is to improve Point Spread Function (PSF) modeling techniques currently used in Single Molecule Localization Microscopy (SMLM), by extending to large samples. First we will introduce the relevant theory to understand the current state-of-the art methods used in SMLM. Next, fluorescence will be introduced, the source of the light we are trying to image. Finally, we will look at the fundamental limit to the maximum resolution we can achieve, the diffraction limit.

1-1 Fluorescence

When a molecule absorbs a photon, it can emit a new photon with a larger wavelength, this effect is called fluorescence [17]. Fluorescence occurs in three steps. First a photon is absorbed, which results in a transition to a higher energy state. Then, due to vibrational losses, the molecule loses some of this energy. Finally, a new photon is emitted and the molecule drops back to the ground state. Because the emitted photon contains less energy than the absorbed photon, it has a longer wavelength. This effect is called the Stokes shift.

In 1962, Green Fluorescent Protein (GFP) has been isolated from a jellyfish. The main breakthrough came in the early 1990 when the gene was sequenced. This allowed GFP to be embedded into the DNA of other organisms. Recently more types of fluorescent chemicals were developed such as the Alexa family of dyes.

Different techniques are used to make sure these fluorescent proteins (fluorophores) are only present in specific regions of interest in an organism. This can be done using genetic modification of the organism of interest. In this case the target protein is modified to include GFP.

Alternatively, organic fluorophores can be used, which have multiple advantages such as smaller size, greater photostability and higher brightness. These organic fluorophores can be coupled to the target protein to label the proteins. This can be done using antibodies, SNAP- or Halo-Tags [18]. The advantage of this technique is that you do not need to genetically modify the sample, and you can use dyes that are more stable.

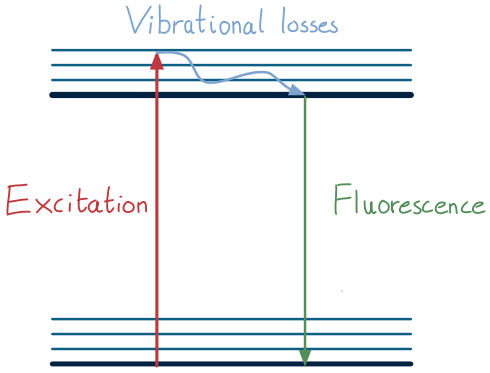


Figure 1-1: Diagram with the different energy states involved in fluorescence.

By using these labeling techniques, a specific area of interest can be labeled and imaged so their workings can be investigated by biologists.

1-2 Diffraction Limit

The resolution (e.g. the size of the smallest detail that can still be resolved) of an optical system is limited. This is true even for a perfect optical system. Due to the wave nature of light, diffraction occurs causing a theoretical limit to the resolution based on the wavelength. This limit is called Abbe's Diffraction Limit and is given by:

$$d = \frac{\lambda}{2n\sin(\theta)} = \frac{\lambda}{2NA}, \tag{1-1}$$

where λ is the wavelength, n the refractive index of the medium around the sample, θ the angle between the marginal and chief ray of the lens, and NA is the numerical aperture. The best case scenario would be a microscope with a high Numerical Aperture (NA) and short wavelength. In practice the limits turn out to be an NA of 1.49, and a wavelength of 405 nm (Alexa-405), which results in a diffraction limit of 136 nm. A similar metric for axial resolution can be found, and is equal to $\lambda/(2NA^2)$ or about 91 nm. See figure 1-2 for an illustration. [19]

1-3 Localization Microscopy

As seen in section 1-2 there is a theoretical limit on the resolution of a microscope. Localization microscopy is a method to improve resolution beyond the diffraction. Localization microscopy is based on the principle that as long as emitters don't overlap it is possible to localize them very precisely. The precision is no longer dictated by the diffraction limit, but is determined by the number of photons collected. A single spot can be localized with precision proportional

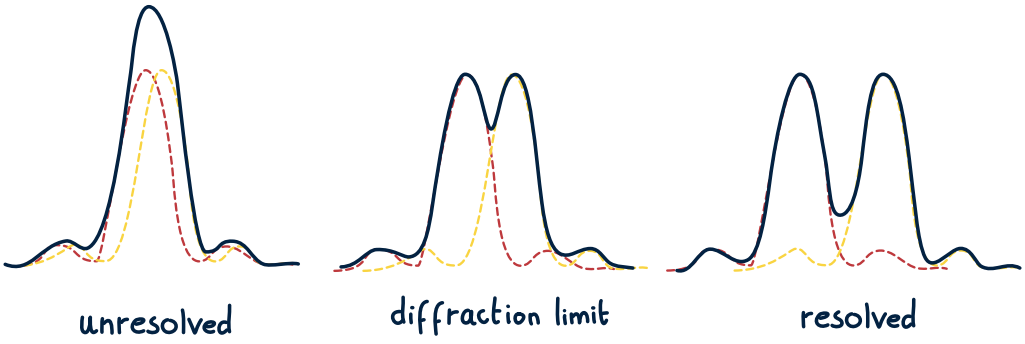


Figure 1-2: Three spacings between sources, unresolvable, exactly at the diffraction limit and not resolved.

to $\sqrt{N}$, where N is the number of photons. A more precise limit to the precision will be presented in section 2-6.

The fluorophore will produce a diffraction-limited spot on the sensor, but by using a model of the PSF, the original emitter position of this spot can be determined. The localization precision is only limited by the number of photons collected. This implies that if more photons are collected from the same source the precision increases. With infinite photons, infinite precision could be achieved. However due to physical limitations, such as drift and photobleaching and the degradation of fluorescent molecules due to photo damage, the precision is limited in practice.

For the localization algorithms to work, an important aspect of localization microscopy is that emitters may not overlap. Therefore, only a small amount of fluorophores should emit light at the same time. By making sure different fluorophores are active at different times a movie can be recorded which contains information about the complete sample. From this movie, all emitter positions can be extracted and turned into the final image. Two main methods are used to get the desired blinking effect, which will be discussed next.

1-3-1 Photo-Activated Localization Microscopy (PALM)

In PhotoActivated Localization Microscopy (PALM) the number of active fluorophores is controlled by using Photo-Activatable Green Fluorescent Protein (PA-GFP). Two lasers are used, one for illumination and one for activation [1, 20].

The activation laser is turned on for a short time, which activates a small part of the fluorophores. These will absorb light from the illumination laser and emit photons. After a while, all the fluorophores are photobleached and no longer emit light. Then, the activation laser is turned on again and the cycle repeats. Each of these activations is captured as a separate frame.

Each frame of the movie is then analyzed. The resolution is limited by the amount of photons emitted by a fluorophores before it bleaches, but can lead to 10x improvement in resolution.

1-3-2 Stochastic Optical Reconstruction Microscopy (STORM)

For Stohastic Optical Reconstruction Microscopy (STORM) the type and activation of the fluorophores is slightly different. Cy5 or Cy3 dyes are used, since they exhibit reversible photo-switching [21]. They can switch for hundreds of cycles, emitting several thousand photons per cycle. Later, other dyes have been used for STORM as well. [22]

STORM uses only one laser for activation and switching back to the dark state. By continuously illuminating the sample with the laser the fluorophores will exhibit stochastic blinking. This stochastic blinking ensures that the total number of active emitters is low.

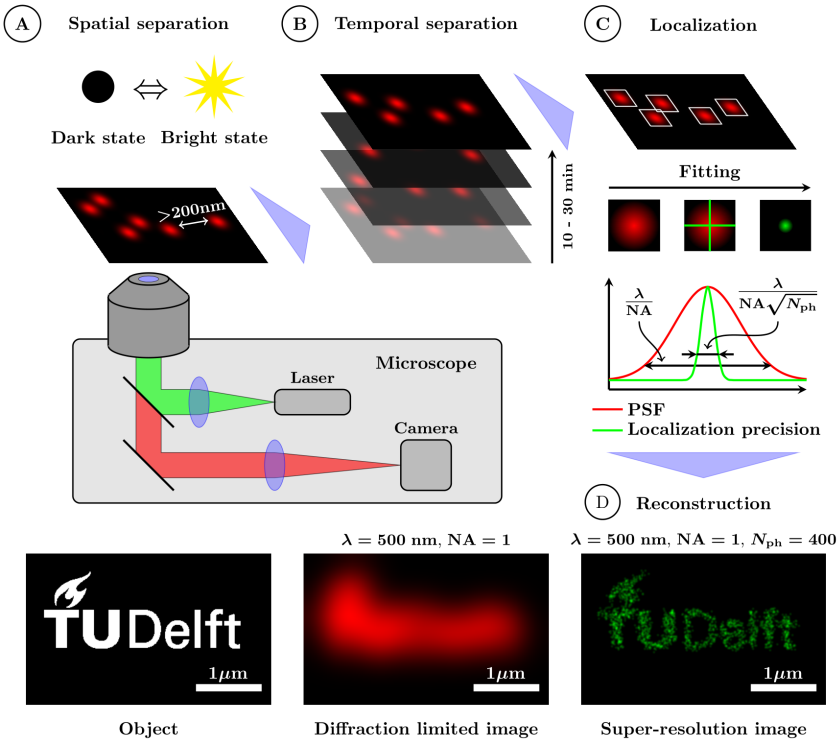


Figure 1-3: Illustration of the principles of PALM/STORM [9]. A) Fluorophores are spatially seperated by blinking. B) By aquiring multiple frames the we get temporal seperation. C) All fluorophores for each frame are localized individually. D) By combining the individual localization the sample can be reconstructed.

1-4 Outlook

In this thesis we would like to answer the question if B-Splines can be used to create accurate and fast models of a PSF. B-Splines will be compared to current state-of-the-art methods, both in localization precision and model accuracy. The methods will be verified on simulations first, and are then put to the test on experimental data. Finally, we will try to extend our B-Splines models to more dimensions, something that is not possible using current methods.

Chapter 2

Theory

2-1 Introduction

This chapter will give an introduction of microscopy techniques. Factors limiting resolution will be discussed and their solution investigated.

First, fluorescence will be introduced, the phenomenon that produces the light that we are trying to image. Then, the diffraction limit is introduced, a physical limit to the smallest details that can be imaged.

Next, to improve resolution beyond the diffraction limit, Single Molecule Localization Microscopy (SMLM) is introduced. The factors limiting resolution in SMLM are discussed. Solutions are presented to increase resolution even further and methods are introduced to measure 3D position or wavelength.

However, those techniques require having a good model of the microscope. Different types of models and approximations are introduced and compared.

2-2 Point Spread Function (PSF)

The Point Spread Function (PSF) is the response of an optical system to a point source. In control theory, this is also called the impulse response of the system. As long as the light entering the microscope is incoherent, it will behave linearly (e.g. no interference occurs). This means that the output of the microscope is a convolution of the PSF with the sample.

2-2-1 Pupil Function

The pupil function is often used to describe how the light is affected when it travels through the microscope.

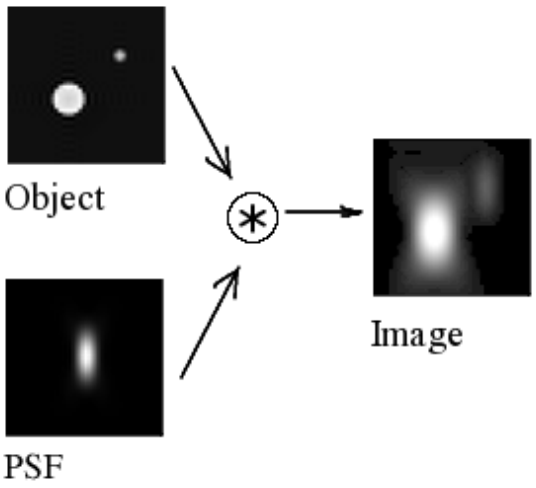


Figure 2-1: The image is the result of the convolution between the PSF and the sample [10].

The Optical Transfer Function (OTF) is defined as the autocorrelation of the Pupil function $P(k_x, k_y)$ [23, Chapter 6]:

$$OTF(k_x, k_y) = P(k_x, k_y) \circledast P(-k_x, -k_y)^*, \tag{2-1}$$

where $\circledast$ denotes the convolution operator. The PSF is defined as the Fourier transform of the OTF:

$$\begin{aligned} \text{PSF(x, y, z)} &= \mathcal{F}\{OTF(k_x, k_y)\} = \mathcal{F}\{P(k_x, k_y) \circledast P(-k_x, -k_y)^*\}, \\ &= \mathcal{F}\{P(k_x, k_y)\} \times \mathcal{F}\{P(k_x, k_y)\}^* = |\mathcal{F}\{P(k_x, k_y)\}|^2. \end{aligned} \tag{2-2}$$

The OTF is only non-zero on a pupil, therefore the 3D Fourier transform can be rewritten as a 2D integral by expressing k_z as function of k_x and k_y [24]:

$$\text{PSF}(x, y, z) = \left| \iint P(k_x, k_y) e^{i2\pi k_z(k_x, k_y) z} e^{i2\pi(k_x x + k_y y)} \, \mathrm{d}k_x \, \mathrm{d}k_y \right|^2, \tag{2-3}$$

where (x, y, z) are the real space coordinates and (k_x, k_y, k_z) the corresponding Fourier space coordinates. $\exp(ik_z(k_x, k_y)z)$ can be considered an out-of-focus blurring factor based on z. The pupil function is a complex valued function describing the change in amplitude and phase at the back focal plane of the objective lens.

2-3 Imaging Model

In the previous chapter the PSF was denoted as $\text{PSF}(x, y, z)$. In practice we want to model the intensity of a specific pixel k. The intensity of pixel is dependent on the pixel number, but also on the (x, y, z) location of the fluorophore, the fluorophore intensity I and the background fluorescence level bg. All these parameters are summarized in the parameter vector θ. The expected number of photons for pixel k is given by:

$$\mu_k(\theta) = \theta_I \cdot \text{PSF}_k(\theta) + \theta_{\text{bg}}, \tag{2-4}$$

where $\text{PSF}_k(\theta)$ indicates the integral of the PSF over the area of pixel k.

2-4 How good is a PSF Model? The χ^2 Test

In the previous section the theoretical response of a microscope to point source has been described. However, we would like to have a statistical test to compare theoretical models to acquired images. Based on the intensity of the image, we would like to know if a statistically significant difference between the experimental image and the theoretical model exists.

For this test the χ^2 can be used:

$$\chi^2 = \sum_{k=1}^{K} \frac{(n_k - \mu_k)^2}{\mu_k}, \tag{2-5}$$

where K is the number of measurements (e.g. number of pixels in an image), n_k is measured intensity for pixel k, and μ_k is the expected intensity for pixel k value from the model. If the measured values are produced by a Poisson process, the expected value and variance for the χ^2 value are given by:

$$\mathbf{E}\left[\chi^2\right] = K, \tag{2-6}$$

$$\mathrm{Var}\left[\chi^2\right] = 2K + \sum_{k=1}^{K} \frac{1}{\mu_k}. \tag{2-7}$$

The level of confidence can be found by comparing the experimental χ^2 to the expected value and variance.

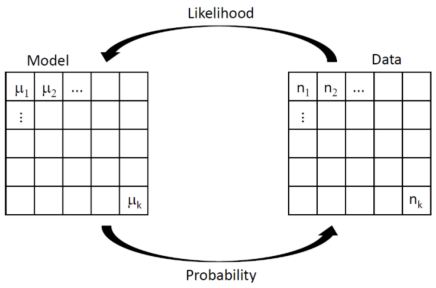


Figure 2-2: Schematic representation of the relation of μ_k, the expected number of photons and n_k the number of measured photons [11]. Each box represents a pixel.

2-5 Maximum Likelihood Estimation

A STORM/PALM experiment will result in a large number of frames taken at different moments in time. All the fluorophores in each frame have to be localized. The most common method is to extract a small Region of Interest (ROI) (a few times larger than the standard deviation of the PSF) based on some initial estimate of the positions (for example by applying a threshold). Section 6-2-2 goes into more detail about ROI selection.

The best estimate of the emitter position can be found using a Maximum Likelihood Estimation (MLE) by using a model of the PSF.

First we can define the likelihood function of a frame [25]:

$$\mathcal{L}(n|\mu(\theta)) = \prod_k P(n_k|\mu_k(\theta)), \tag{2-8}$$

where $n = (n_1, n_2, \ldots, n_K)$ is the observed image with pixel intensities n_k, $\mu(\theta) = (\mu_1(\theta)), \mu_2(\theta), \ldots, \mu_K(\theta))$ is the expected intensity from the imaging model (including noise) given the parameters θ, and $P(n_k|\mu_k(\theta))$ is the conditional probability of observing n_k given $\mu_k(\theta)$. This conditional probability is determined by the noise model. In case of a Poisson model the probability is given by:

$$P(n_k|\mu_k(\theta)) = \frac{\mu_k(\theta)^{n_k}}{n_k!} e^{-\mu_k(\theta)}, \tag{2-9}$$

where $\mu_k(\theta)$ is the expected number of photons according to the imaging model given the parameters θ. The MLE fit then consists of finding the θ that maximizes $\mathcal{L}$:

$$\hat{\theta} = \arg\max_{\theta} \ \mathcal{L}(n|f(\theta)). \tag{2-10}$$

To make computations easier, we can also normalize the function by dividing by the maximum possible likelihood $\mathcal{L}(n|n)$:

$$\hat{\theta} = \arg\min_{\theta} \ -2\ln\left(\frac{\mathcal{L}(n|f(\theta))}{\mathcal{L}(n|n)}\right). \tag{2-11}$$

By combining equations (2-8), (2-9) and (2-11) we arrive at:

$$\hat{\theta} = \arg\min_{\theta} \ 2\sum_{i=1}^{k}(\mu_i(\theta) - n_i) - 2\sum_{i=1, n_i\neq 0}^{K} n_i \ln(\frac{\mu_i(\theta)}{n_i}). \tag{2-12}$$

With the likelihood statistic defined, an optimization algorithm such as Levenberg-Marquardt [26, 27] or Newton-Raphson can be used to find the best estimate of the parameters θ.

2-6 Cramér-Rao Lower Bound (CRLB)

Using MLE the parameters θ can be estimated. However, there is a theoretical limit to the precision with which these parameters can be retrieved [28].

This limit is related to the amount of information present in the PSF. If there is a vector of relevant parameters θ, usually (x, y, z, I, bg), the entry (i, j) of Fisher information matrix is then defined as:

$$I_{ij}(\theta) := -\mathbf{E}\left[\frac{\partial^2}{\partial\theta_i \partial\theta_j} \log f(n;\theta)\right], \tag{2-13}$$

where $f(n;\theta)$ is the likelihood function that image n has the parameters θ. If $T(n)$ is an unbiased estimator of θ the lower bound on the covariance is given by the Cramér-Rao bound:

$$\text{cov}(T(n)) \geq I(\theta)^{-1}. \tag{2-14}$$

The measurements from a detector that is only limited by the arrival of photons and not the readout can be modeled as a Poisson process. For a single pixel k, the expected number of

photons is defined as $\mu_k(\theta)$, if the process is Poisson distributed. The entry (i, j) of the Fisher information matrix for the whole image is then given by:

$$I_{ij}(\theta) = \sum_k \frac{1}{\mu_k(\theta)} \frac{\partial \mu_k(\theta)}{\partial \theta_i} \frac{\partial \mu_k(\theta)}{\partial \theta_j}. \tag{2-15}$$

2-7 Root Mean Square Error

In statistics we can describe the precision and accuracy of an estimator using the variance (or standard deviation) and bias respectively. But sometimes ,we need a metric to compare two different estimators. The Root Mean Square Error (RMSE) of the estimator is defined as:

$$\text{RMSE} = \sqrt{\frac{\sum_{n=1}^{N} (\hat{\theta}^n - \theta^n)^2}{N}}, \tag{2-16}$$

where N is the number of images. The RMSE can also be defined in terms of bias and standard deviation:

$$\text{RMSE} = \sqrt{\text{bias}^2 + \text{std}^2}. \tag{2-17}$$

This means that, for an unbiased estimator, the RMSE is equal to the standard deviation.

2-8 Phase Aberrations

The pupil function P from equation (2-3) can be written as:

$$P(\vec{\rho}) = A(\vec{\rho}) e^{iW(\vec{\rho})}, \tag{2-18}$$

where $A(\vec{\rho})$ is the amplitude aberration function, and $W(\vec{\rho})$ is the phase aberration function. $A(\vec{\rho})$ depends on the aperture. for a microscope, the pupil is circular. The effect of imperfections in the microscope or optical elements changing the phase of the light, can be described by the phase aberration function $W(\vec{\rho})$.

2-8-1 Zernike Polynomials

We would like to be able to easily represent phase aberrations often found in practice. Zernike polynomials can be used as a basis for the aberration function $W(\vec{\rho})$, since they are orthogonal on the unit disk [29]. The Zernike polynomials are identified by two positive integers, n and m as follows:

$$Z_n^m(\rho, \phi) := R_n^m(\rho) \cos(m\phi) \quad \text{(even)}, \tag{2-19}$$

$$Z_n^{-m}(\rho, \phi) := R_n^m(\rho) \sin(m\phi) \quad \text{(odd)}. \tag{2-20}$$

$$R_n^m(\rho) := \sum_{k=0}^{\frac{n-m}{2}} \frac{(-1)^k (n-k)!}{k!(\frac{n+m}{2} - k)!(\frac{n-m}{2} - k)!} \rho^{n-2k}. \tag{2-21}$$

The aberration function can then be written as a linear combination of Zernike polynomials:

$$W(\vec{\rho}) = \sum_n \sum_m k_n^m Z_n^m(\vec{\rho}). \tag{2-22}$$

The first few of these Zernike polynomials describe classical aberrations. Z_0^0 is piston, Z_1^{-1} is Y-Tilt, Z_1^2 is X-Tilt, Z_2^{-2} is oblique astigmatism, etc. For most models, only phase aberrations are considered, but it is also possible to write $A_{\text{apl}}(\vec{\rho})$ as a linear combination of Zernike polynomials.

It is customary to normalize the magnitude of the aberrations by the wavelength. This means the magnitudes are expressed with unit of mλ.

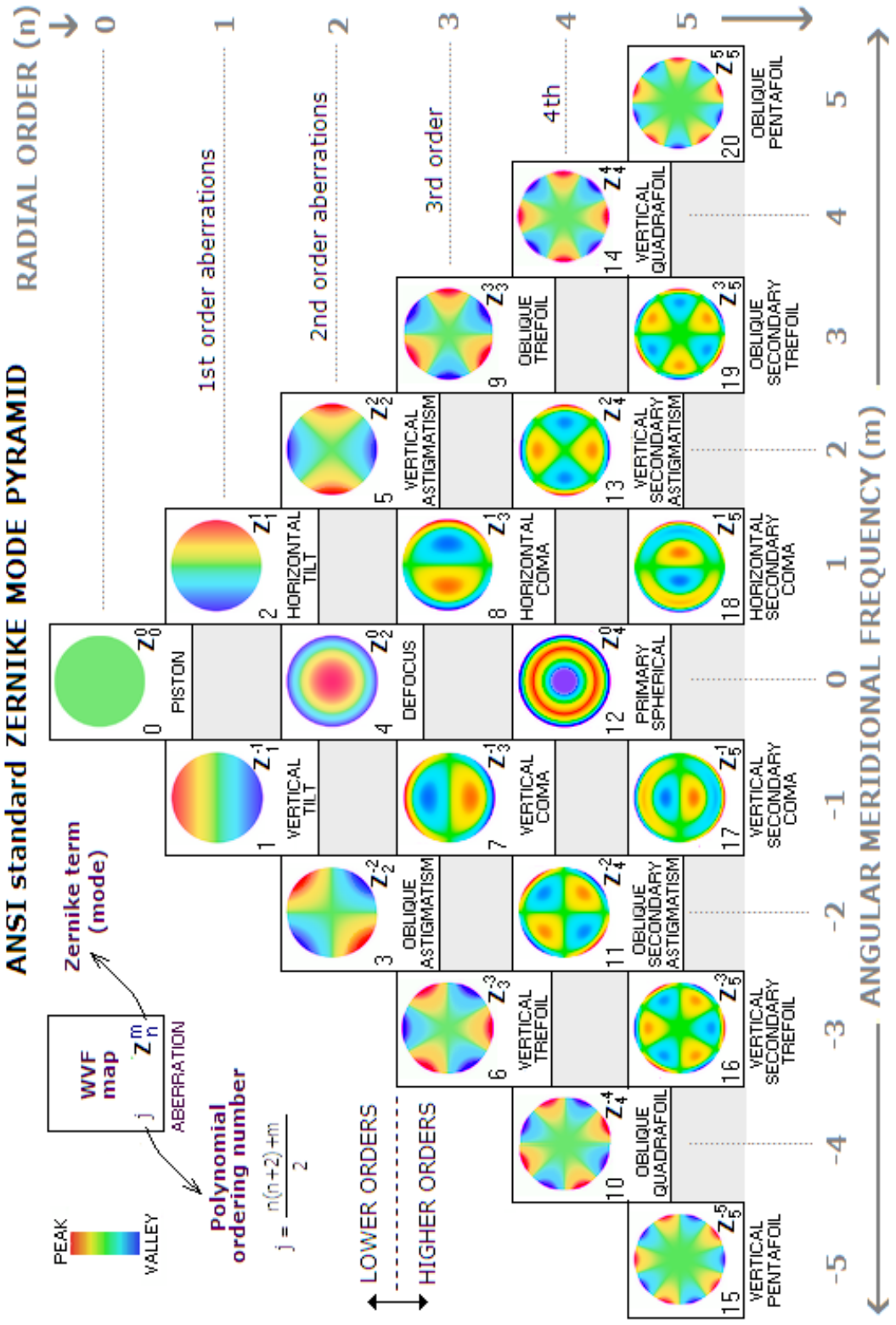


Figure 2-3: Phase of the pupil function for the first 21 Zernike polynomial [12].

2-9 PSF Engineering

To allow reconstruction of a 3D structure from a 2D image, information about the axial position of the emitters has to be present in the image. By introducing a known aberration, a PSF can be designed which optimizes accuracy or information about the z-position. By inserting certain optical elements in the microscope it is indeed possible to influence the PSF, such as a cylindrical lens, an Spatial Light Modulator (SLM) or a Digital Micromirror Device (DMD).

2-9-1 Astigmatic

Adding a cylindrical lens adds an astigmatic effect. This gives the PSF an elliptical shape. The z-position can be recovered from the relative width of the PSF in the x and y direction. However, for larger axial positions, the PSF loses its elliptical shape. The Cramér-Rao Lower Bound (CRLB) with respect to z increases, and therefore the axial range is limited to a few hundred nm.

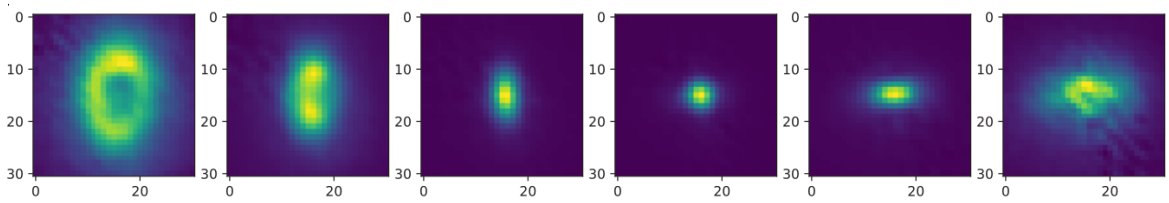


Figure 2-4: Astigmatic PSF. Note that for larger axial position the PSF loses the elliptical shape.

2-9-2 Double Helix

The double helix PSF consists of two spots that rotate around a center when the z-position changes. The axial position can be calculated from the relative angle of these two spots. [30] The desired axial range can be controlled by tuning the rotation of the two spots.

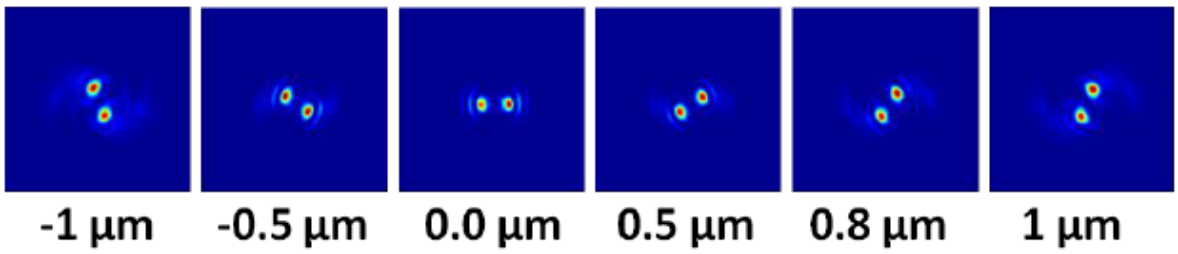


Figure 2-5: Double Helix PSF. With changing axial position the angle between the two spots changes [13].

2-9-3 Tetrapod

In section 2-6 the maximum precision of a certain PSF was defined as the CRLB. To create an optimal PSF, an optimization can be performed to improve the CRLB with respect to the

desired parameter (e.g. z). Each pixel of an SLM can be treated as a free variable, or Zernike modes of the phase aberration function can be used to limit the degrees of freedom. [28]

The Tetrapod is a "Saddle-Point" PSF, and is the result of this optimization. Up to this point, PSFs only had a usable range of about 3 µm, the Tetrapod was optimized to have much larger usable range. For all tested z-ranges (from 2 µm to 20 µm) the algorithm produces a PSF with two distinct lobes. The distance between the lobes changes with z-position and is rotated 90° above and below the focal plane. [14]

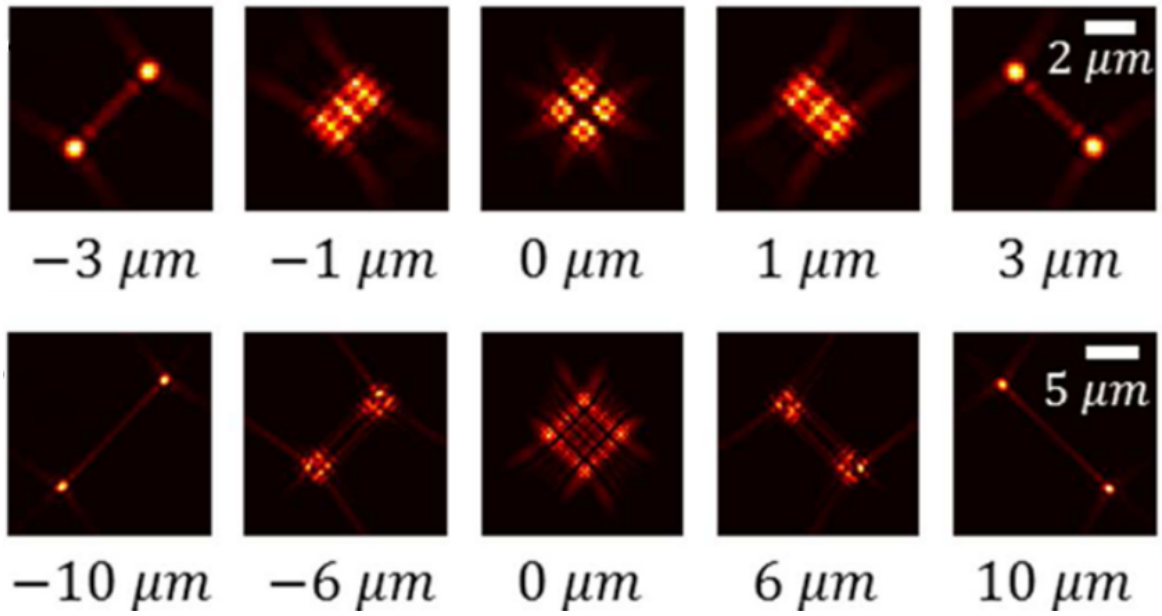


Figure 2-6: Tetrapod PSFs, one optimized for small axial range (top, $\pm$ 3 µm), and one for large axial range (bottom, $\pm$ 10 µm) [14].

2-9-4 3D + λ

Unlike the sensor in your camera or smartphone, a sensor used in microscopy has no color filters in order to maximize the number of photons captured by the sensor. However, this means the resulting images contain no information about the wavelength of the light. Color information can be retrieved by adding color filters in front of the camera, but this also limits the number of photons and therefore localization precision. Ideally, we would like to recover the wavelength based on the gray-scale images. Having information about the wavelength allows simultaneous labeling with two different colored dyes, for example to study the interaction between two parts of the sample.

With the right PSF, it is possible to recover wavelength [9]. These "3D + λ" PSFs have two elliptical lobes. The relative width of the ellipsoids can be used to recover z, and the distance between the two lobes is a measure for λ. These PFSs show precision below 10 nm in x, y and λ, and below 20 nm in z over an axial range of 1 µm.

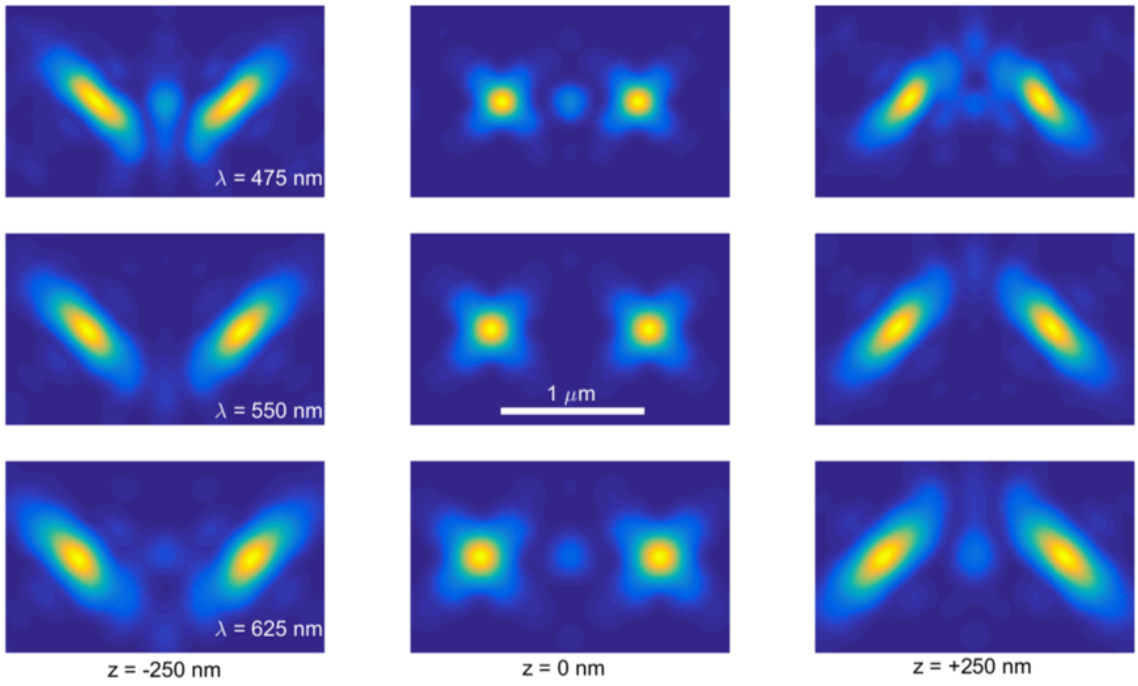


Figure 2-7: 3D + λ PSF for three different wavelengths, 475 nm, 550 nm and 625 nm, and three different z-positions [15]. When the wavelength increases, so does the distance between the spots.

2-10 PSF Approximations

The previous section described PSFs designed to recover as much information as possible. We would like to use these PSFs in a localization algorithm and therefore, we need a model.

One of the most important factors in the localization accuracy is the accuracy of the model used to represent the PSF in the MLE fit. The optimization algorithm requires thousands of evaluations of the model. Therefore, it needs to be reasonably fast to compute as well.

Furthermore, to be able to use more efficient optimization algorithm such as Levenberg-Marquardt and Newton-Raphson, the derivative of the PSF also needs to be known.

2-10-1 2D Gaussian

The PSF can be approximated by a 2D Gaussian function:

$$\text{PSF}(x,y) = \frac{1}{2\pi\sigma_0^2} e^{-\frac{1}{2\sigma_0^2}\left[(x-x_0)^2+(y-y_0)^2\right]}, \tag{2-23}$$

with σ_0 the width of the Gaussian distribution and (x_0, y_0) the position of the emitter. To get the expected value for a pixel, we must integrate over the surface area of the pixel and take into account the intensity (θ_I) and the background (θ_{bg}) [31]:

$$\mu_k = \theta_I \Delta E(x_k - \theta_x)\Delta E(y_k - \theta_y) + \theta_{bg}, \tag{2-24}$$

with

$$\Delta E(u) = \frac{1}{2}\left[\text{erf}\left(\frac{u+\frac{1}{2}}{\sqrt{2}\sigma_0}\right) - \text{erf}\left(\frac{u-\frac{1}{2}}{\sqrt{2}\sigma_0}\right)\right], \tag{2-25}$$

where (x_k, y_k) are the pixel coordinates and (θ_x, θ_y) the location of the center of the PSF.

2-10-2 3D Gaussian

When the z-position of the PSF changes it becomes defocussed and the size of the PSF will increase. The Gaussian PSF can be extended to model this by varying the standard deviation with z:

$$\sigma(z) = \sigma_0\sqrt{1 + \frac{z^2}{d} + A\frac{z^3}{d} + B\frac{z^4}{d}}, \tag{2-26}$$

where A and B are empirical constants, σ_0 is the in focus standard deviation, d is the depth of focus (a constant for a given objective). In practice it turns out that B is almost zero.

As discussed in section 2-9-1, a common method to increase axial resolution is to introduce a cylindrical lens, which will give the PSF an ellipsoid shape. The PSF model from equation (2-23) can be extended to describe the astigmatic effect [32]:

$$\text{PSF}(x,y) = \frac{1}{2\pi\sigma_x(\theta_z)\sigma_y(\theta_z)} e^{-\frac{(x-\theta_x)^2}{2\sigma_x(\theta_z)^2} - \frac{(y-\theta_y)^2}{2\sigma_y(\theta_z)^2}}, \tag{2-27}$$

where θ_x, θ_y and θ_z are the emitter x, y and z position. $\sigma_x(\theta_z)$ and $\sigma_y(\theta_x)$ are given by:

$$\sigma_x(z) = \sigma_{0_x}\sqrt{1 + \frac{(z-\gamma)^2}{d^2} + A_x\frac{(z-\gamma)^3}{d^2} + B_x\frac{(z-\gamma)^4}{d^2}}, \tag{2-28}$$

$$\sigma_y(z) = \sigma_{0_y}\sqrt{1 + \frac{(z-\gamma)^2}{d^2} + A_y\frac{(z-\gamma)^3}{d^2} + B_y\frac{(z-\gamma)^4}{d^2}}. \tag{2-29}$$

2-10-3 Vectorial Model

The scalar based model from section 2-2 does not take into account the freely rotating dipole of the emitter. Every dipole orientation experiences a different pupil modulation. The Vectorial model correctly describes the freely rotating dipole by extending the model [15].

$$\mathrm{PSF}(\vec{r}) = \frac{N}{3} \sum_{l=x,y} \sum_{j=x,y,z} \left| \iint P(\vec{\rho}) q_{lj}(\vec{\rho}) e^{i\vec{k}(\vec{\rho})\cdot\vec{r}} \, \mathrm{d}^2\rho \right|^2, \tag{2-30}$$

where $\vec{k}(\vec{\rho})$ is the wave vector, $q_{lj}(x, y, z)$ represents the polarization vector components with electric field component $l = x, y$ in the image plane proportional to the emission dipole component $j = x, y, z$.

Even though the Vectorial model takes into account the freely rotating dipole, even this model does not perfectly describes the microscope behavior. After calibrating all aberrations present in the optical system to below 20 mλ, a statistically significant difference is present between the acquired images and the Vectorial model for an intensity of 2.4×10^4 photons and a background of 19 photons/pixel [9]. Also, the Vectorial model requires Fourier transforms to evaluate, which makes it relatively slow when used in an iterative method such as MLE.

2-10-4 C-Splines

Cubic splines (C-Splines) were successfully used to model a PSF [3, 4]. Splines allow fitting a model to arbitrary PSF shapes, such as the double helix or tetrapod. Also the first and second derivative of the spline functions can be computed analytically. The downside is a slightly higher computational load compared to a Gaussian model. The expression for the C-Splines is given by:

$$f_{i,j,k}(x, y, z) = \sum_{m=0}^{3} \sum_{n=0}^{3} \sum_{p=0}^{3} a_{i,j,k,l,m,p} \left(\frac{x - x_i}{\Delta x} \right)^m \left(\frac{y - y_j}{\Delta y} \right)^n \left(\frac{z - z_k}{\Delta z} \right)^p, \tag{2-31}$$

with Δx and Δy the pixel size, Δz the step size in z direction. x_i, y_j and z_k the start positions of the voxel (i, j, k) in the x, y and z direction. $a_{i,j,k,l,m,p}$ is the spline coefficient, of which 64 are used per voxel. See section 6-3 for the procedure to determine the spline coefficients.

2-11 Comparision between Models

In the previous section, different models were discussed. We would like to use a model that is a simple as possible, but not if that leads to loss of precision or introduction of a bias.

For example, in [2] it is shown that using a Gaussian approximation systematic errors are introduced. When the dipole is fixed, a defocus aberration of 72 mλ is present and about 10^4 photons are used, a systematic error of up to 40 nm is present.

Therefore, the Gaussian approximation cannot be used if very high localization precision is required. The Vectorial model does not have these issues, but is very computationally

expensive. Thus, another approximation method is needed that is fast to compute, but more accurate than the Gaussian approximations.

Both requirements are satisfied by the C-Spline models. However, those approximations have a few downsides. First, the number of spline coefficients is very higher, which requires lots of calibration data. Second, the model is not very flexible and is only feasible to use on a rectangular 3D domain. The next chapter will look into modeling the PSF using B-Spline, which use less parameters and are more flexible.

Chapter 3

B-Splines

3-1 Introduction

Splines are a very flexible tool to model datasets. The C-Splines as described in section 2-10-4 have some shortcomings: firstly, they can't fit scattered data, and therefore require interpolation and shifting of the measurement data to the voxel grid. Secondly, they only work on rectangular domains. This is fine for 2D and 3D modeling, but later we will use the B-Splines to model variations of the Point Spread Function (PSF) over the field of view. The data is then no longer spread out over a uniform grid. The theory about B-splines in this chapter is a summary of Coen de Visser's work on using B-splines to fit the aerodynamic model of an airplane [5].

A methodology was proposed to model arbitrary datasets using multivariate simplex splines. These simplex splines are defined on geometric entities called simplices. For a point in a simplex the value of the spline can be evaluated using a linear combination of Bernstein basis polynomials.

The splines on each simplex are joined together using continuity conditions to span the whole domain. This configuration is called a triangulation and the density of simplices can be used to match the local system complexity.

3-2 Simplices

An n-dimensional simplex, or n-simplex is the convex hull of a set of $n+1$ non-degenerate points (e.g. the convex hull is of dimension n). The 1-simplex is a line, the 2-simplex a triangle and a 3-simplex a tetrahedron. $\mathcal{V}_t$ is the tuple[1] of the n-simplex's vertices:

$$\mathcal{V}_t := (\mathbf{v}_{p_0}, \mathbf{v}_{p_1}, \ldots, \mathbf{v}_{p_n}). \tag{3-1}$$

[1] A tuple is an ordered list of elements. The order of the vertices becomes relevant later on, so a set cannot be used.

The simplex t is then defined as the convex hull of $\mathcal{V}_t$:

$$t := \langle \mathcal{V}_t \rangle, \tag{3-2}$$

where $\langle \cdot \rangle$ denotes the convex hull operation. A useful entity related to an n-simplex is the normalized vertex matrix $\mathbf{A}_t$. This matrix contains the vertices of the simplex relative to the first vertex:

$$\mathbf{A}_t = \begin{pmatrix} \mathbf{v}_{p_1} - \mathbf{v}_{p_0} & \mathbf{v}_{p_2} - \mathbf{v}_{p_0} & \dots & \mathbf{v}_{p_n} - \mathbf{v}_{p_0} \end{pmatrix}. \tag{3-3}$$

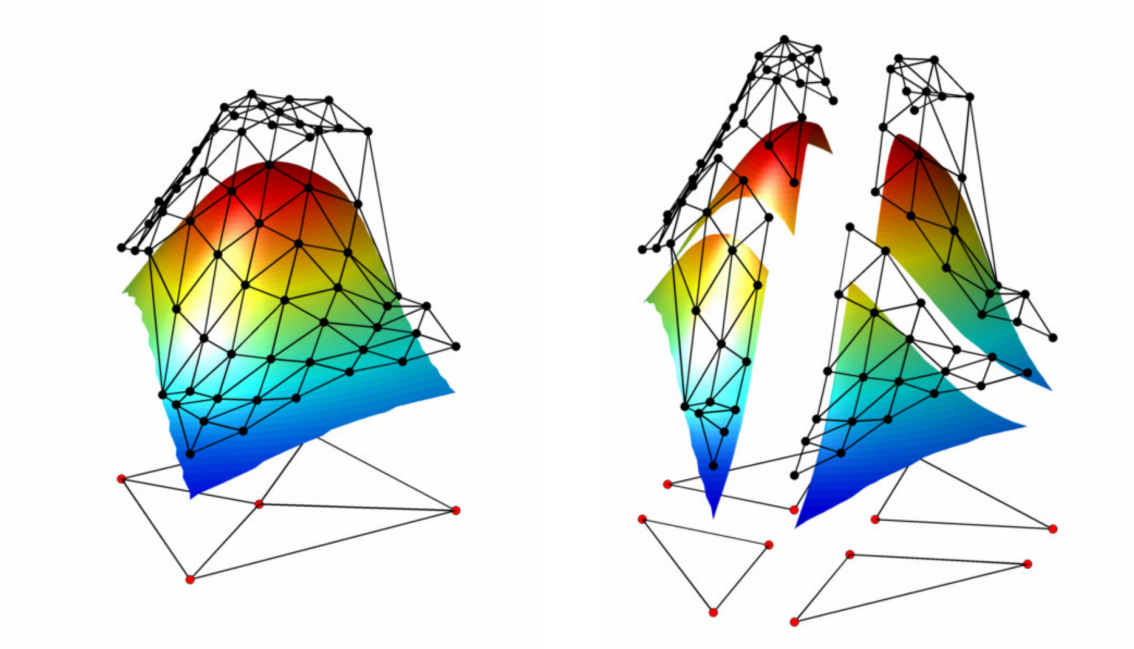


Figure 3-1: Example of a simplex spline with four simplices (from[5]). On the right side the four different simplices are separated for clarity. The black grid on top of the graph shows the B-net, the black dots indicate the physical location of the B-coefficients.

3-3 Barycentric Coordinates

To be able to express quantities about points in a simplex, we could use Cartesian coordinates. However, this would require a lot of coordinate tranformations. Therefore, to express the location of a point in a simplex, a new coordinate system is introduced: barycentric coordinates. The barycentric coordinate system represents locations with respect to the vertices of the simplex.

The relationship between the Euclidian coordinate $\mathbf{x} = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n$, the simplex vertices $\mathbf{v}_{p_i}$ and the barycentric coordinate $\mathbf{b} = (b_0, b_1, \dots, b_n) \in \mathbb{R}^{n+1}$ is given by:

$$\mathbf{x} = \sum_{i=1}^{n} b_i \mathbf{v}_{p_i}, \tag{3-4}$$

since $\sum b_i = 1$, the degrees of freedom is reduced by one. The following expressions fully define $\mathbf{b}$:

$$\begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} = \mathbf{A}_t^{-1} \cdot (\mathbf{x} - \mathbf{v}_{p_0}), \tag{3-5}$$

$$b_0 = 1 - \sum_{i=1}^{n} b_i. \tag{3-6}$$

3-4 Triangulations

To create a full simplex spline, the splines on individual simplices have to be combined. The configuration of simplices to span the whole domain is called the triangulation. The triangulation has a large effect on the quality of the model. First, metrics will be introduced to measure the quality of a triangulation. The next sections will describe different methods of generating a triangulation.

3-4-1 Simplex Metrics

To distinguish between 'good' and 'bad' triangulations some metrics have to be defined. The SDP metric is introduced in [5] and is based on the number of spline coefficients per simplex. The other metrics are introduced in [33].

SRLA Location and radius of circumsphere center. The center of the circumsphere should be inside of the simplex

SRSC Ratio radius circumsphere and shortest ridge. The radius of the circumsphere should at most be two times larger than the shortest edge of the simplex

SMA Minimum angle. The minimum angle of the simplex should be at least 14°

SDP Minimum data points. The simplex should contain at least as many datapoints as spline coefficients. The number of spline coefficients is given $\hat{d} = \binom{d+n}{d}$ where d is spline degree and n the dimension.

The first three metrics determine the simplex geometry, but don't require anything about the data. When a simplex geometry is chosen, the fourth requirement becomes very important. This requirement states that the system is overdetermined; there have to be more datapoints than there are spline coefficients that need to be determined. Very often the number of datapoints is limited, and this requirement will pose a limit on the resolution of the triangulation.

3-4-2 Delaunay Triangulation

The Delaunay triangulation generates the triangulation which maximizes the minimum angle of all the simplices [34, Chapter 9]. It ensures each circumsphere only contains vertices belonging to that simplex.

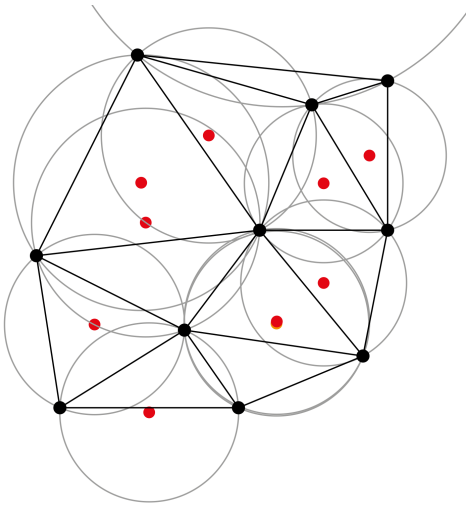


Figure 3-2: Delaunay triangulation

3-4-3 Type I/II Triangulations

Type I and Type II triangulations do not depend on the location of the data points. A single template triangulation of an n-cube is repeated to fill the whole domain. The difference between Type I (figure 3-3) and Type II (figure 3-4) is that Type II uses an additional vertex at the center of the n-cube. These triangulations can also be used to span non rectangular domains by removing unneeded simplices.

Type I and Type II triangulations are very simple to generate and have good simplex metrics. However, their use is limited in dimensions larger than 5. The number of simplices per cube is $n!$ [35]. For example, the 8-cube contains 256 vertices, and its Type I triangulation contains 33313 simplices. Even to fit a 1^{st} order spline, 9 data points per simplex are needed, or about 3 million per 8-cube.

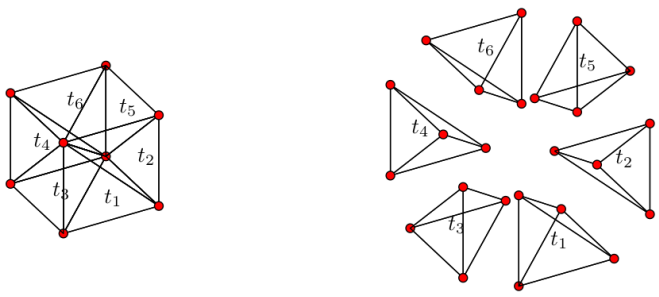


Figure 3-3: Type I triangulation of a cube [5].

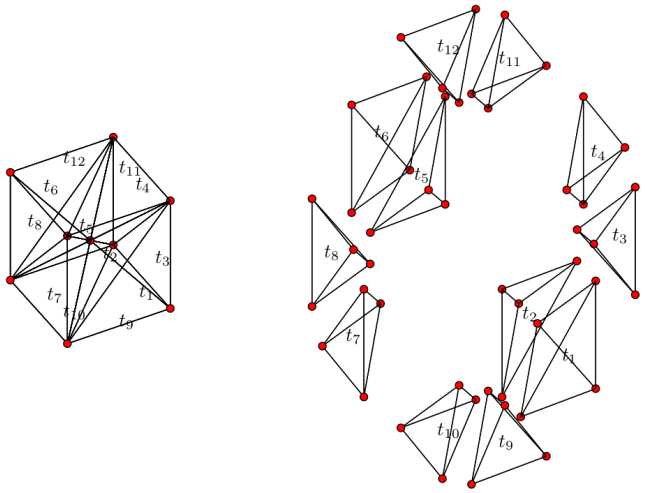


Figure 3-4: Type II triangulation of a cube [5].

3-5 Bernstein Basis Polynomials

Each simplex contains a simplex polynomial, e.g. a linear combination of Bernstein basis polynomials. These polynomials form a basis on the simplex. This section will derive these basis polynomials. These polynomials can be defined for a given degree d.

For the barycentric coordinate $\mathbf{b} = (b_0, b_1, \ldots, b_n)$, the multinomial theorem states:

$$(b_0 + b_1 + \ldots + b_n)^d = \sum_{\kappa_0+\kappa_1\ldots\kappa_n=d} \frac{d!}{\kappa_0!\kappa_1!\ldots\kappa_n!} b_o^{\kappa_0} b_1^{\kappa_1} \ldots b_n^{\kappa_n}. \tag{3-7}$$

The sum on the right hand side sums over all possible multi-indices of length n with sum d. The multi index κ of dimension n is defined as:

$$\kappa := (\kappa_0, \kappa_1, \ldots, \kappa_n) \in \mathbb{N}^{n+1}, \tag{3-8}$$

with the 1-norm:

$$|\kappa| = \kappa_0 + \kappa_1 + \ldots + \kappa_n = d, \quad d \geq 0, \tag{3-9}$$

and factorial:

$$\kappa! = \kappa_0!\kappa_1 \ldots \kappa_n!. \tag{3-10}$$

Using equations (3-9) and (3-10), equation (3-7) can be simplified to:

$$(b_0 + b_1 + \ldots + b_n)^d = \sum_{|\kappa|=d} \frac{d!}{\kappa!} \mathbf{b}^\kappa. \tag{3-11}$$

Thus, the Bernstein basis polynomial is then given by:

$$B_\kappa^d(\mathbf{b}) := \frac{d!}{\kappa!} \mathbf{b}^\kappa, \tag{3-12}$$

since $\mathbf{b}$ is a barycentric coordinate (where all coordinates sum to 1), equations (3-11) and (3-12) can be combined into:

$$1 = \sum_{|\kappa|=d} B_\kappa^d(\mathbf{b}) \tag{3-13}$$

With the basis polynomial defined the value of a spline, at coordinate $\mathbf{b}$ with B-coefficients $\mathbf{c}$, can be defined as:

$$p(\mathbf{b}) := \sum_{|\kappa|=d} c_k B_\kappa^d(\mathbf{b}). \tag{3-14}$$

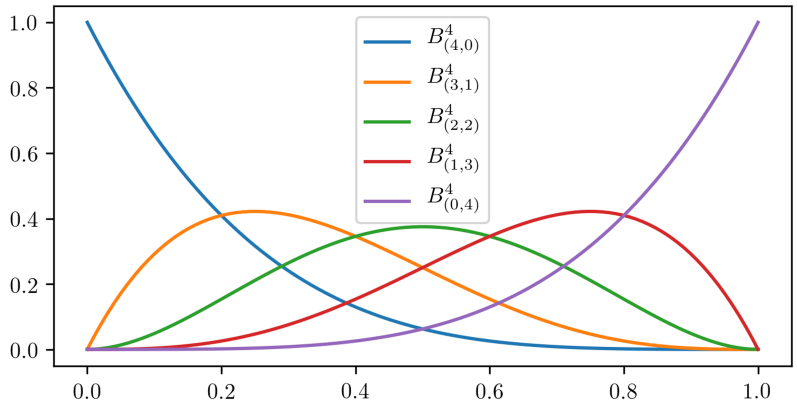


Figure 3-5: Basis functions for a 4th degree spline in 1D for for a simplex with nodes $v_{p_0} = (0)$ and $v_{p_1} = (1)$.

3-6 Continuity Conditions

Inside of a simplex the B-form polynomial is a differentiable function. A d^{th} degree B-form polynomial has a continuity of C^d within the spline. However the continuity between different simplices C^r is chosen when fitting the B-coefficients. Higher continuity comes at a cost, by increasing the continuity order the degrees of freedom of the splines are lowered. This lowered approximation power can be represented by the semi-degree or relative degree:

$$\delta := d - r - 1. \tag{3-15}$$

3-7 Coefficient Fitting

The previous sections explained how to build and evaluate a B-Spline with given coefficients. However, we still need to fit the coefficients to our dataset. A linear regression method will be used to determine the coefficients, given some dataset. For each datapoint i we define the measurement $y(i)$ at location $\mathbf{x}(i)$:

$$y(i) = f(\mathbf{x}(i)) + r(i), \qquad i = 1, 2, \dots N, y(i) \in \mathbb{R}^1, \tag{3-16}$$

where f is the model (in our case the PSF), $r(i)$ a residual error between the model and the measurements, and N the number of measurements.

For our regression we replace the function f by our B-Spline. Recall equation (3-14), where the value of the spline was defined for a given simplex. To define an expression which is valid over the whole triangulation, we introduce the membership operator:

$$\delta_{j,k(i)} = \begin{cases} 1, & \text{if } j = k(i), \\ 0, & \text{otherwise,} \end{cases} \tag{3-17}$$

where $k(i)$ is an index function which produces the simplex index which contains the coordinate $\mathbf{x}(i)$. This means that the membership operator $\delta_{j,k(i)}$ is equal to one when the data point $\mathbf{x}(i)$ lies inside simplex j. With the membership operator defined we can define the regression model:

$$y(i) = \sum_{j=1}^{J} \left(\delta_{j,k(i)} \sum_{|\kappa|=d} c_k B_\kappa^d(\mathbf{b}(i)) \right) + r(i), \tag{3-18}$$

where J is the number of simplices. To complete the regression model, we would like a matrix form which contains all the measurements. First, we need the $\mathbf{B}_{t_j}^d$, the vector of basis polynomial terms for the simplex t_j:

$$\mathbf{B}_{t_j}^d(i) = [B_\kappa^d(\mathbf{b}(i))]_{|\kappa|=d} \in \mathbb{R}^{1\times\hat{d}}. \tag{3-19}$$

To ensure consistency, the possible values of the multi-index κ will be lexicographically sorted. Equation (3-18) can now be simplified to:

$$y(i) = \sum_{j=1}^{J} \left(\delta_{j,k(i)} \mathbf{B}_{t_j}^d(i) \cdot \mathbf{c}^{t_j} \right) + r(i), \tag{3-20}$$

where $\mathbf{c}^{t_j} \in \mathbb{R}^{\hat{d}\times 1}$ is the vector of spline coefficients for the simplex t_j.

$\hat{d}$ is defined as the total number of permutations of κ with length n and sum d:

$$\hat{d} := \binom{d+n}{d} = \frac{(d+n)!}{n!d!}. \tag{3-21}$$

We also need a matrix form of the membership operator for a data point i:

$$\mathbf{D}_{t_j}(i) = \begin{bmatrix} \delta_{j,k(i)} & 0 & \dots & 0 \\ 0 & \delta_{j,k(i)} & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & \delta_{j,k(i)} \end{bmatrix} \in \mathbb{R}^{\hat{d}\times\hat{d}}. \tag{3-22}$$

The vector with basis functions for all simplices can be defined as:

$$\mathbf{B}^d(i) = \begin{bmatrix} \mathbf{B}_{t_1}^d(i) & \mathbf{B}_{t_2}^d(i) & \dots & \mathbf{B}_{t_J}^d(i) \end{bmatrix} \in \mathbb{R}^{1\times J\cdot\hat{d}}, \tag{3-23}$$

and similar for the spline coefficients:

$$\mathbf{c} = \begin{bmatrix} \mathbf{c}^{t_1} & \mathbf{c}^{t_2} & \dots & \mathbf{c}^{t_J} \end{bmatrix}^\intercal \in \mathbb{R}^{J\cdot\hat{d}\times 1}. \tag{3-24}$$

We also define a block matrix from of the membership operator that is valid for all simplices for a given data point:

$$\mathbf{D}(i) = \begin{bmatrix} \mathbf{D}_{t_1} & 0 & \dots & 0 \\ 0 & \mathbf{D}_{t_2} & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \dots & 0 & \mathbf{D}_{t_J} \end{bmatrix} \in \mathbb{R}^{(J\cdot\hat{d})\times(J\cdot\hat{d})}. \tag{3-25}$$

Using equations (3-23) to (3-25) we can write the regression as:

$$y(i) = \mathbf{B}^d(i) \cdot \mathbf{D}(i) \cdot \mathbf{c} + r(i). \tag{3-26}$$

We can define the row i of the regression matrix X as $X(i) = \mathbf{B}^d(i) \cdot \mathbf{D}(i)$, $\mathbf{Y}$ as the vector containing all the observations $y(i)$, and $\mathbf{r}$ as the vector of all the residuals $r(i)$. This leads to the following equation for all the data-points:

$$\mathbf{Y} = \mathbf{X}\mathbf{c} + \mathbf{r}, \qquad \mathbf{Y} \in \mathbb{R}^{N\times 1}. \tag{3-27}$$

For this analysis we assume $\mathbf{r}$ has zero mean and $\mathrm{Cov}(\mathbf{r}) = \mathbf{\Sigma}$. If there are no continuity continuity constraints equation (3-27) can be solved by:

$$\hat{\mathbf{c}} = \arg\min_{\mathbf{c}} \ \frac{1}{2}(\mathbf{Y} - \mathbf{X}\mathbf{c})^\intercal \Sigma^{-1} (\mathbf{Y} - \mathbf{X}\mathbf{c}). \tag{3-28}$$

In case of continuity conditions, we need to add a constraint to this optimization problem. The continuity conditions can be represented using the smoothness matrix $\mathbf{H}$, which adds the constraint $\mathbf{H}\mathbf{c} = \mathbf{0}$. The construction of the smoothness matrix is discussed in[5, Section 3.3]. The equality constraint is then added to the original problem by using the Lagrange multiplier λ:

$$\hat{\mathbf{c}} = \arg\min_{\mathbf{c}} \ \frac{1}{2}(\mathbf{Y} - \mathbf{X}\mathbf{c})^\intercal \Sigma^{-1} (\mathbf{Y} - \mathbf{X}\mathbf{c}) + \lambda^\intercal(\mathbf{H}\mathbf{c}). \tag{3-29}$$

3-8 Derivatives

It is possible to compute the derivatives for a Simplex B-Spline analytically. This derivative can be pre-computed and results in a new set of spline coefficients. This new spline has a degree of one less than the original spline. The continuity order also decreases with one when taking the derivative.

3-9 Conclusion

In this chapter we introduced the theory behind simplex B-Splines. Different tuning parameters were identified, degree, triangulation and continuity order. In the next chapter we will use the B-Splines to model a PSF, and look at the effect of each of the tuning parameters.

Chapter 4

B-Spline Model Fitting

4-1 Introduction

In this chapter we will look at the results when fitting a B-Spline on a simulated Point Spread Function (PSF). PSFs will be generated using the Vectorial model, which will be used to fit a B-Spline model. Afterwards the fitted model will be compared to the input data to see how well they match. Later chapters will deal with fitting on experimental data.

The B-Spline model has different parameters that affect the model quality, but also affect the required number of data points for calibration. In this chapter, the effect of model parameters such as triangulation, spline degree and continuity order will be discussed, and recommended parameters will be selected.

4-2 Procedure

To assess the model quality realistic PSFs have to be generated. For this the Vectorial model (section 2-10-3) will be used. For most simulations, an astigmatic aberration will be added to simulate the PSFs used in some 3D localization techniques such as 3D STORM (section 2-9-1).

This Vectorial model is used to generate z-stacks of 151 frames with a size of 14x14 pixels. To make sure there is enough data to correctly fit the model, 9 stacks are generated with all combinations of 0, 1/3 and 2/3 pixel offsets in x and y. A vertical astigmatism (Z_2^2) with a magnitude of $50\,\mathrm{m}\lambda$ was used.

To simulate the real data acquisition process, background fluorescence is added and Poisson noise is applied. A background intensity of 20 photons per pixels was used. The bead intensity was chosen relatively high at 10^7 photons per frame in the z-stack. We do not want to choose a too low intensity, or this will limit the maximum reachable model quality. This high intensity is therefore a trade-off between simulating real world data, and limiting the potential fit quality due to a low signal to noise ratio.

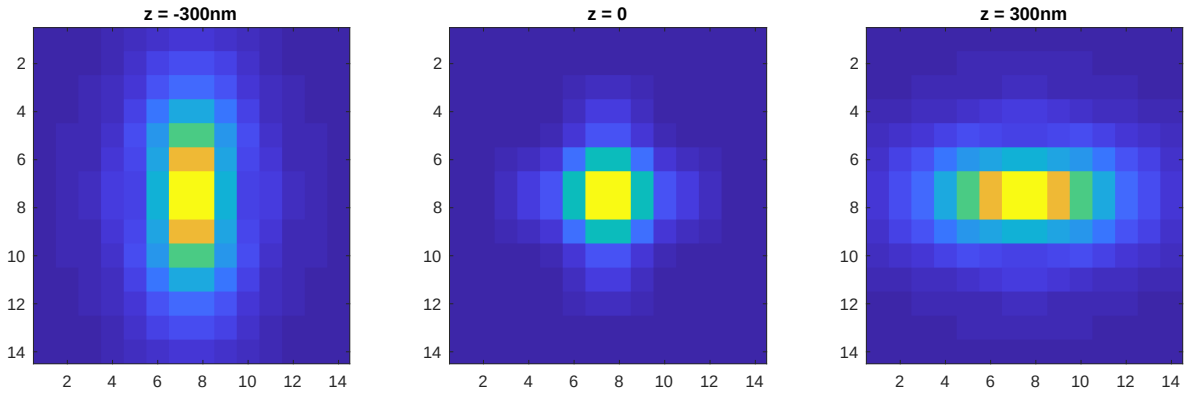
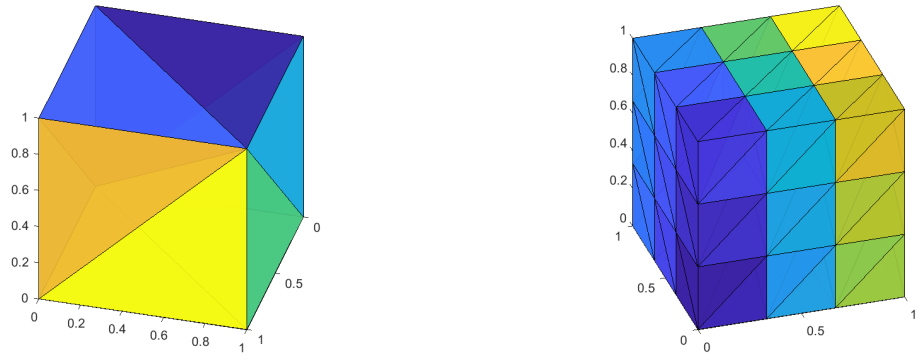


Figure 4-1: Astigmatic PSF used for benchmarks. The complete z-stack is $14 \times 14 \times 151$ pixels. In this figure frames 1, 76 and 151 are shown.

A Type I triangulation is made as described in section 3-4-3. The whole domain is then divided in M cubes in each dimension. This means the domain is triangulated using $d! \cdot M^3$ simplices, where d is the dimension.



(a) $M = 1$ **(b)** $M = 3$

Figure 4-2: Examples of triangulations with 1 and 3 cubes. In the left figure, the 6 simplices making up the cube can be seen.

4-3 Model Quality Assessment

To compare the three models ,B-Spline, C-Spline and 3D Gaussian, a χ^2 test is used. This test gives an indication until which intensity the approximated model is indistinguishable from the theoretical Vectorial model.

In section 2-4, the χ^2 test was introduced. For an image with given dimension,we can calculate the expected value and the variance of the χ^2 value. The measurement data with different intensities is simulated using the vectorial model, a background of 1 is added and then Poisson noise is applied. For each intensity, 64 different realizations are generated. Then, for each of these realizations the χ^2 value is computed for each of the three models.

This χ^2 value can then be compared to their expected value and variance to determine at which intensity a given model is no longer valid. Figure 4-3 shows an example of the χ^2 value versus the number of photons per frame. This shows the model is valid until about 5×10^4 photons.

When the background increases, the model will be valid with a higher intensity. The background intensity of 1 therefore produces a lower bound for the intensity where the model is still statistically valid.

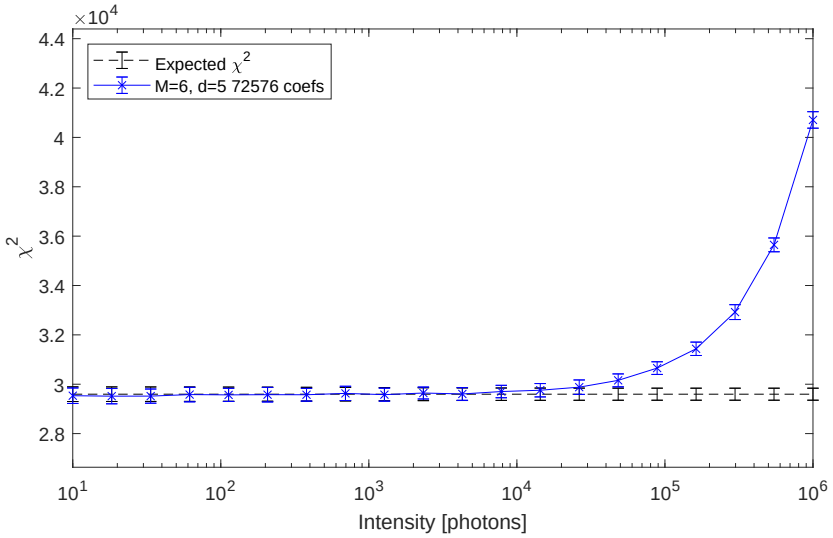


Figure 4-3: χ^2 plot with six cubes in each dimension, spline degree 5 and 1st order continuity. When the photon count exceed about 5×10^4 photons the B-Spline model doesn't match the Vectorial model anymore.

4-4 B-spline Model Parameters

The parameters influencing the model quality are the triangulation, the spline degree and the continuity order. This section will discuss each of these parameters, and study the influence on the model quality.

4-4-1 Triangulation (M)

The first parameter is the triangulation. There a infintely many possible triangulations. However, we would like triangulations that can be easily generated up to dimensions of 5. At these low dimensions, Type I still have a reasonable number of simplices (the number of simplices is equal to $n!$). The triangulations can be generated by repeating a template triangulation across the domain. This template is the n-dimensional cube, or n-cube, with unit length vertices.

To generate the triangulation of the n-cube, we can use a simple algorithm. Each simplex will contain the vertex $(0, 0, 0, \ldots)$ and $(1, 1, 1, \ldots)$. The simplices can be generated by computing all possible paths between these two edges along the outside of the cube. This can be done by

taking a unit step in any direction, as long as the sum of the coordinates increases by exactly one each step.

In 3D, we start with $(0, 0, 0)$. Then, there are three possible edges $(0, 0, 1), (0, 1, 0), (1, 0, 0)$. For $(0, 0, 1)$, there are two possible steps left, $(0, 1, 1), (1, 0, 1)$. The same can be done for the other two edges after the first step. The third step of the path will always end in $(1, 1, 1)$. By using this method we can generate all $3 \cdot 2 \cdot 1 = 3! = 6$ paths. The simplices are then formed by taking the convex hull of these paths. See appendix A-1 for a Python script to generate triangulations for arbitrary dimensions.

In this research only Type I triangulations are considered, which significantly reduces the number of possibilities. However, the number of cubes used to subdivide the domain can still be varied.

To show the effect of the triangulation on the χ^2 splines with different triangulations were fitted. The degree was kept constant at 5. The results are shown in figure 4-4. As expected, the intensity where χ^2 shows a significant difference increases with the number of cubes.

Pitfalls

The number of cubes has direct influence on the number of simplices and therefore on the number of coefficients. Thus, we expect the model quality to improve when the number off cubes is increased. However, when the number of simplices is increased the number of data points required for a good fit also increases.

Remember from section 3-4-1 that we need at least a certain number of data points per simplex to get a good fit. For example, when the dimension is 3 and the degree is 5 we require $\binom{d+n}{d} = \binom{8}{5} = 56$ points. If M is the number of cubes, the total number of required data points is $\cdot\binom{d+n}{d} \cdot n! \cdot M^3$. If we again take $n = 3$ and $d = 5$ and in this case $M = 6$ we require 72572 data points assuming that the data points are evenly distributed. See table 4-2 for the number of spline coefficients for different values of M and d. The simulation generates 266364 evenly distributed datapoints, which should be sufficient for all degrees and and number of cubes in the table.

However, we will see that if the number of coefficients gets too close to this number, the fit will still fail. So, this relationship should be seen as a rule of thumb.

4-4-2 Degree (d)

To show the effect of spline degree on model quality, B-Splines were fitted with degrees between 1 and 7. Just as with the triangulation, the model quality increases with increasing degree. See figure 4-5.

Pitfalls

When trying to fit splines that have a too low relative degree, they can't accurately represent the data. When this happens they splines tend to show "wiggles" around the simplex edges.

Table 4-1: Number of spline coefficients for varying degree and number of cubes.

		Number of cubes (M)					
		1	2	3	4	5	6
Degree (n)	1	24	192	648	1536	3000	5184
	2	60	480	1620	3840	7500	12 960
	3	120	960	3240	7680	15 000	25 920
	4	210	1680	5670	13 440	26 250	45 360
	5	336	2688	9072	21 504	42 000	72 576
	6	504	4032	13 608	32 256	63 000	108 864
	7	720	5760	19 440	46 080	90 000	155 520
	8	990	7920	26 730	63 360	123 750	213 840
	9	1320	10 560	35 640	84 480	165 000	285 120
	10	1716	13 728	46 332	109 824	214 500	370 656

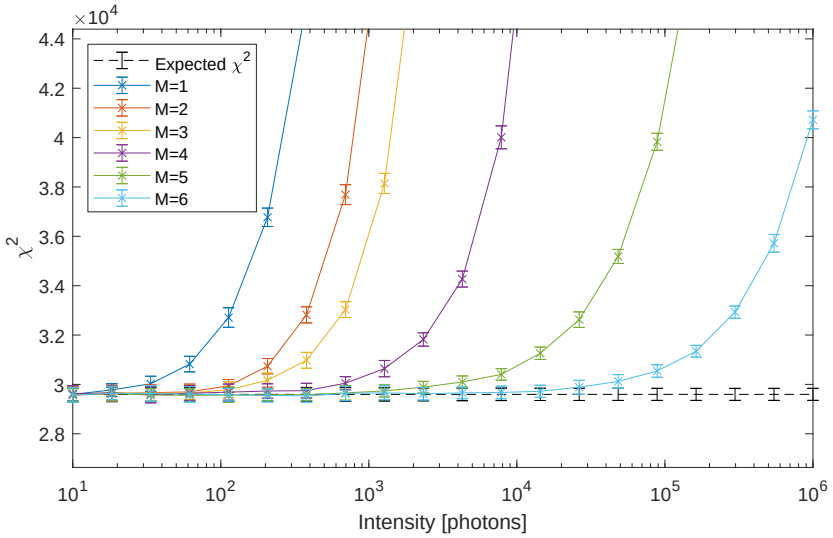


Figure 4-4: χ^2 plot with varying number of cubes in each dimension, spline degree 5 and 1st order continuity.

This especially happens near the edges of the Region of Interest (ROI) where the intensity is lower.

These large deviations show up when plotting the χ^2 values per pixel (figure 4-6), but they can also be spotted by comparing the fitted spline to the reference data along a line through z. In figure 4-7 these effects occur with spline degree 5, but disappear when the degree is increased to 7, because the approximating power of the spline increases.

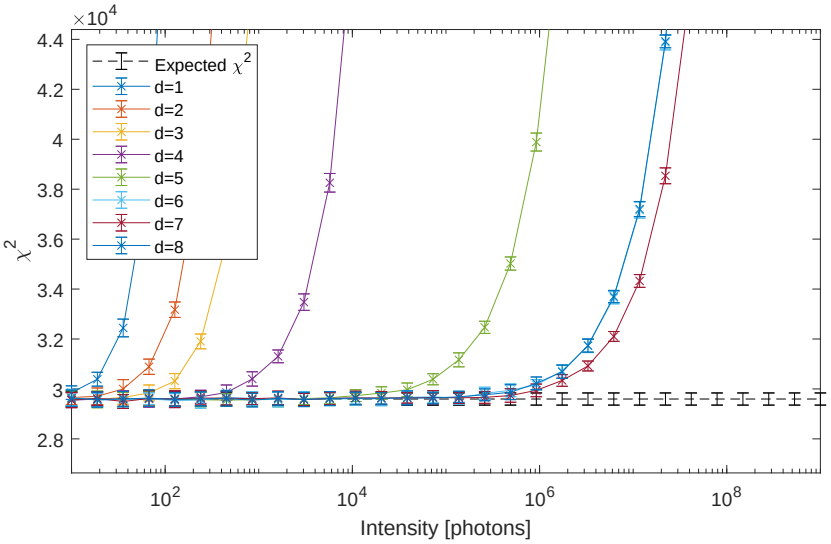


Figure 4-5: χ^2 plot with six cubes in each dimension, varying spline degree 1st order continuity. Note that above degree 7 the model quality starts to decrease due over fitting.

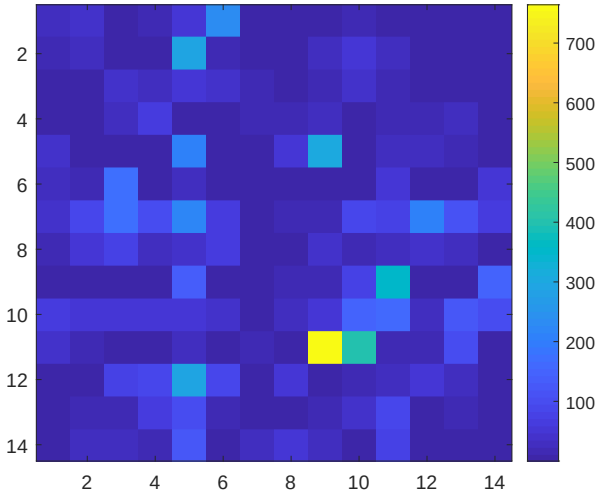


Figure 4-6: χ^2 map of a B-Spline model containing wiggles. This is one slice of z-stack, where the brightness of the pixel indicated the χ^2 value. Wiggles show up as a very large χ^2 value, as can be seen in the bottom center. This example has degree 3 and 6 cubes, z $= -188\,\mathrm{nm}$.

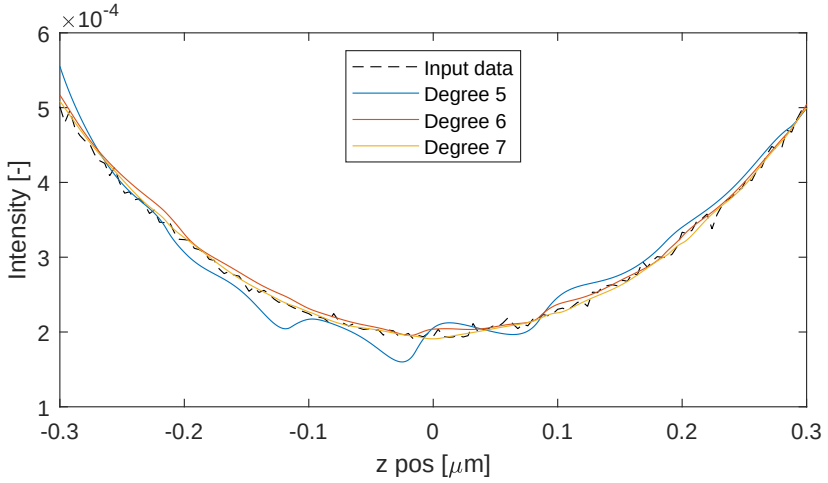


Figure 4-7: Plot of intensity along the z-axis for $x = 1$ and $y = 1$. Note with degree 5 wiggles are still visible.

4-4-3 Continuity Order C^r

The continuity order describes how many derivatives will be continuous between simplices. Increasing the continuity order decreases the expressiveness of the spline, but makes it more smooth.

Figure 4-8 shows the impact of the continuity order on the model quality. Model quality decreases with increasing continuity order, except for 0^{th} order which is worse than 1^{st} order. This is probably caused by over fitting the noise.

Pitfalls

The expressiveness of the spline can also be formulated as the relative degree (equation (3-15)), the difference between spline degree and continuity order. This means that increasing the continuity order effectively decreases the spline degree.

Since the B-spline model will be used in an optimization algorithm that uses the first derivative, a continuity order of at least 1 is needed to guarantee a stable fit [36].

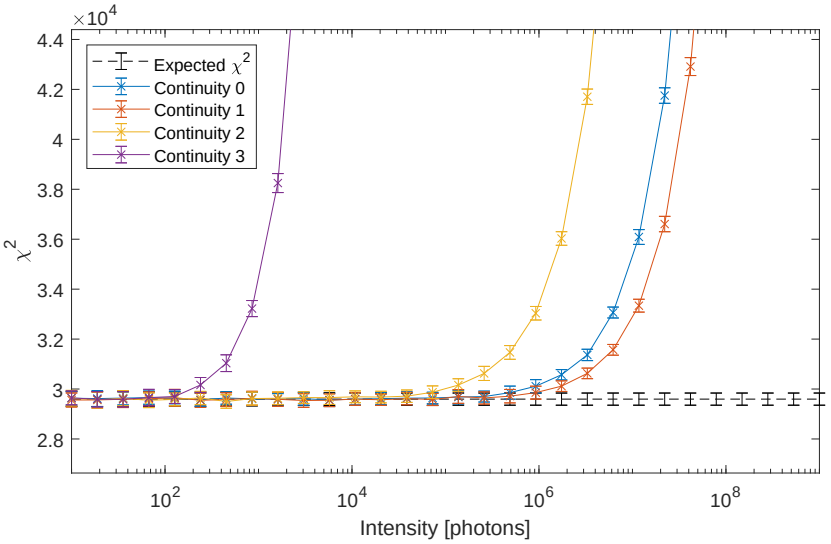


Figure 4-8: χ^2 plot with 5 cubes in each dimension, degree 7. Continuity varying between 0 and 3. The best model quality is reached with continuity 1. With continuity 0 the model probably over fits the noise.

4-5 Recommended Parameters

As seen the previous sections there are a certain number of degrees of freedom in choosing the degree, triangulation and continuity order. A continuity order of 1 is recommended. Lower is not possible due to fitter stability, and higher will decrease model quality.

Table 4-2 shows the effect of spline degree and number of cubes on the fit quality of the model. A higher photon count where a significant difference occurs indicates a better model. We can

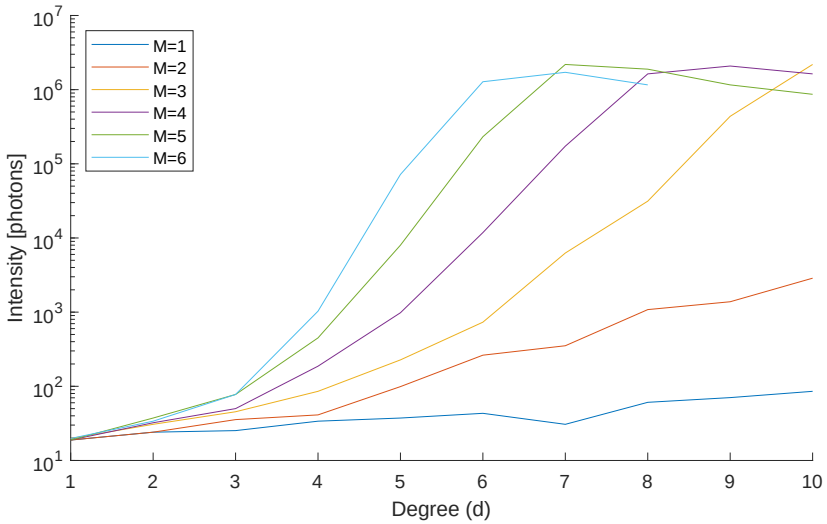


Figure 4-9: Intensity where the χ^2 value is larger than 3 times the standard deviation of the expected value.

also note that over fitting occurs when the degree becomes too large. Thens the model will try to fit the noise in the input data, which will reduce model quality.

To choose the other parameters, we would have to look at the available datapoints from calibration measurements. The number of datapoints is a direct upper limit to the number of coefficients. For a given number of coefficients we would like to find the set of parameters (degree, triangulation) that gives the best model quality while resulting in a number of spline coefficients less or equal to the number of datapoints available.

For example, if we have 1000 datapoints available, the number of options is limited. For $M = 1$ we can choose any degree between 1 and 8, for $M = 2$, we can choose degree 1, 2 or 3. For $M = 3$ we can only choose degree 1. We would like to choose the set of parameters for the highest model quality. In this case this would be degree 8 and 1 cube.

Table 4-3 shows the optimal parameters for a maximum number of spline coefficients. For the given amount of calibration data and intensity, $d = 7$ and $M = 5$ are the optimal parameters. Those parameters are used throughout the rest of the benchmarks.

Table 4-2: Intensity where a significant difference occurs according to the χ^2 test. This occurs when the χ^2 value is larger than 3 times the standard deviation of the expected value. For each number of cubes, the degree with the highest intensity is denoted in bold.

		Number of cubes (M)					
		1	2	3	4	5	6
Degree (d)	1	19	19	20	19	19	20
	2	24	24	31	32	37	34
	3	25	36	45	50	78	78
	4	34	41	86	187	450	1030
	5	37	99	227	981	7997	71 854
	6	43	263	732	11 815	231 737	1 278 266
	7	31	352	6266	172 926	**2 186 268**	**1 712 998**
	8	61	1082	31 350	1 631 427	1 888 580	1 159 425
	9	70	1381	436 975	**2 082 160**	1 159 425	-
	10	**86**	**2871**	**2 186 268**	1 631 427	865 180	-

Table 4-3: Optimal parameters for a given maximum of spline coefficients.

Max. coefficients	Degree (n)	Cubes (M)	Intensity
1000	8	1	61
10 000	8	2	1082
100 000	7	5	2 186 268

4-6 Comparison to Other Approximations

In the previous sections, we compared the model quality for different parameter options of the B-Spline model. Now that we have found the optimal parameters, we would like to compare the final model to other PSF approximations. This section will compare the B-Splines to the 3D Gaussians and C-Splines.

4-6-1 Models

Gaussian 3D

The first model used for comparison is the Gaussian model from equation (2-27). A maximum likelihood fit is used to find the optimal parameters $(x, y, \sigma_x, \sigma_y)$ for each frame in the z-stack.

C-Spline

The second model used for comparison is the C-Spline model [3]. Their open source software can be found on Github[1], and a Python package can be downloaded. The same 9 z-stacks as

[1] https://github.com/ZhuangLab/storm-analysis

used for the B-Spline calibration are placed into a tiff file with very high intensity. Then using the command line tools from the Python package a C-Spline model is fitted on this data.

The toolbox applies some regularization to the data to improve robustness when using experimental data. However, this decreases the model quality when relatively clean data is used. Therefore the regularization was switched off.

Then, a fitter and evaluator was built into our own SMLM toolbox to evaluate the C-Spline model for different (x, y, z) positions.

4-6-2 Chi Square

To determine the difference in model quality between Guassian, C-Spline and B-Spline, again a χ^2 test is used. The results are plotted in figure 4-10. The quality of the C-Spline model is larger than that of the Gaussian, but they are both outperformed by the B-Spline. The B-Spline model also has about 20 times less parameters.

When plotting the error between the C-Spline and the reference PSF, a lot of structure can be seen. At this point it is unknown which part of the C-Spline fitting process causes these errors.

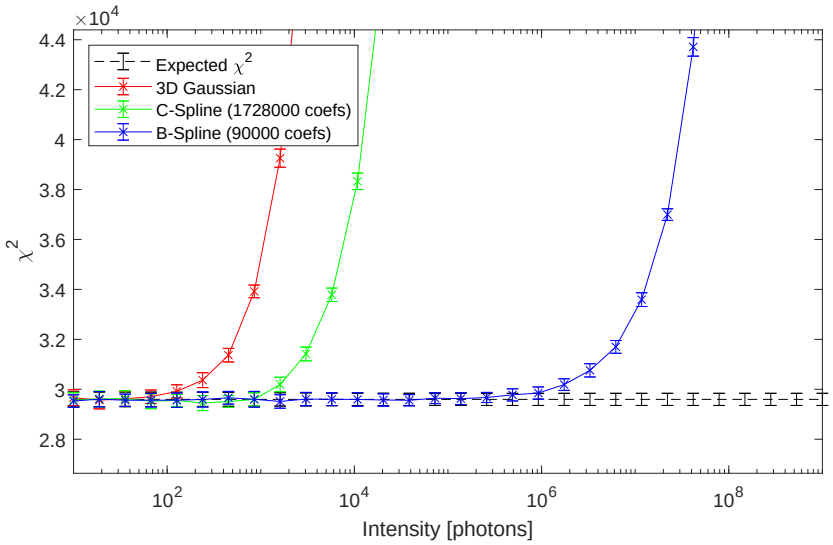


Figure 4-10: χ^2 plot comparing B-spline, C-Spline and Guassian model. The B-Spline model outperforms the C-Sline model even though it uses 19 times less parameters

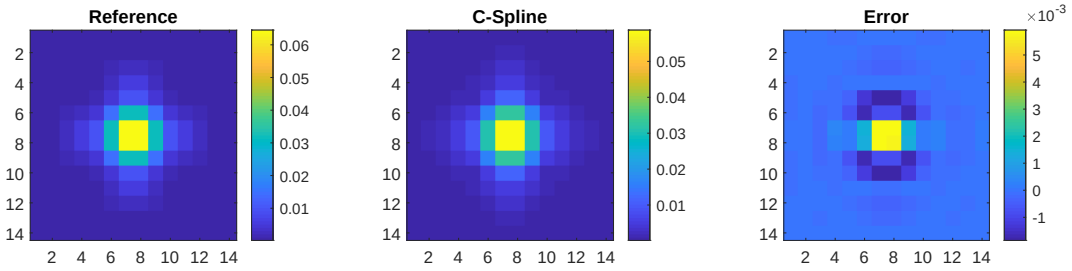


Figure 4-11: Difference plot between C-Spline and Reference PSF

4-7 Conclusion

In this chapter, we fitted a B-Spline model to a PSF. We used a very accurate physics based model, the Vectorial model, to generate the input data. The real imaging process was simulated by adding a background and applying Poisson noise.

The B-Spline model has different parameters such as spline degree and triangulation influencing the number of coefficients and fit quality. A statistical test was used to measure the impact on these parameters on the model quality. Optimal parameters were suggested for different number of calibration measurements.

These preliminary results look very promising: B-Splines can very accurately model a PSF, using much less parameters than current state of the art methods. Using less parameters makes calibration easier, by not requiring lots of measurements or other methods to generate datapoints such as regularization followed by up-sampling.

Another benefit of B-Splines is that the input data does not have to be shifted to a fixed grid before the data points can be used. Because less pre-processing of the data is required, the image quality is not deteriorated before fitting.

Figure 4-12 shows a comparison between the Vectorial model, B-Splines, C-Splines and the 3D Gaussian model. Only the Gaussian model can be visually distinguished from the Vectorial model. Even though it is hard the see visually, the χ^2 for the C-Spline model indicates a statistically significant difference.

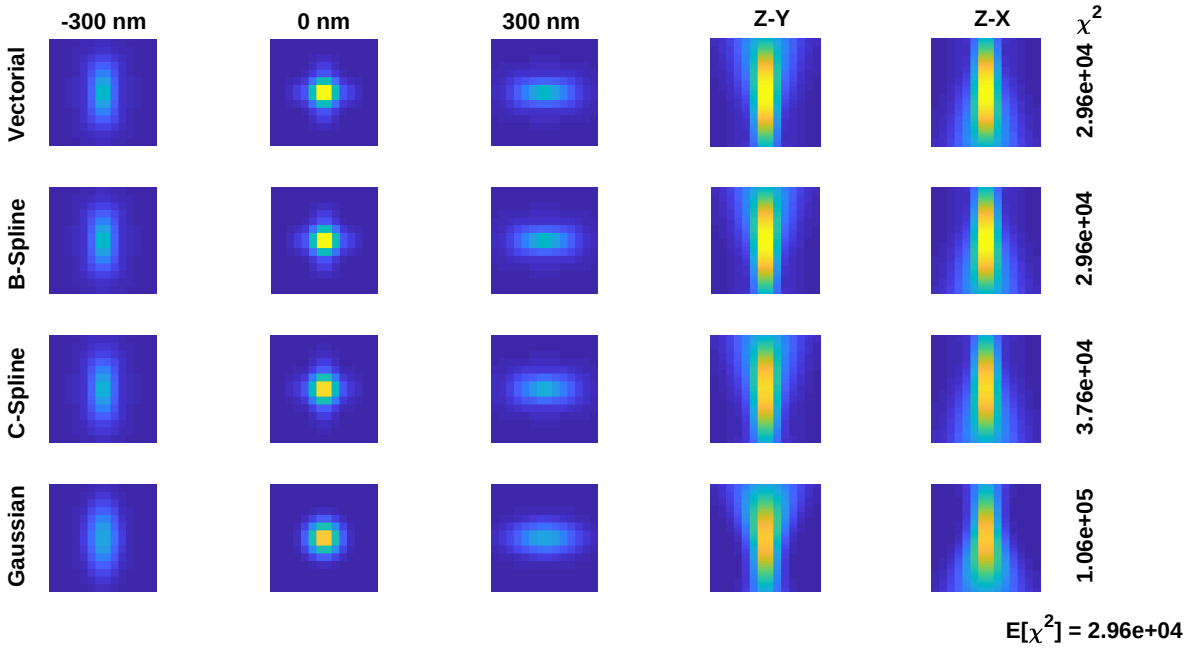


Figure 4-12: Comparison between the different models. The intensity was 10^4 photons, and a background intensity of 1 photon per pixel. At this intensity the χ^2 value for the C-Spline and the Gaussian model are clearly higher than expected value.

Chapter 5

Fitter Implementation

5-1 Introduction

The previous chapter explored the parameter space for the optimal choice of B-Spline model parameters. The metric that was used to measure model quality was the χ^2 value.

In this chapter, we will implement a fitter and measure the fitter performance by looking at accuracy and precision. This means the B-Splines have to be implemented in a Maximum Likelihood Estimation (MLE) algorithm. This chapter will discuss some practical details relevant when building a fast implementation of a B-Spline model on a GPU.

5-2 Implementation Details

This section contains some important implementation details to ensure numerical stability and fast performance. This includes methods for the barycentric coordinate transform and the simplex selection.

5-2-1 Barycentric Coordinate Transform

One of the steps that needs be executed many times is a transformation from Cartesian coordinates to Barycentric coordinates. In section 3-3, an expression for this conversion was derived:

$$\mathbf{A_t} \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix} = \mathbf{x} - \mathbf{v}_{p_0}. \tag{5-1}$$

When evaluating the model, the simplex vertices remain constant. So, for each simplex the matrix $\mathbf{A_t}$ is constant. This means we could pre-compute the inverse of this matrix, and

compute $A_t^{-1}(\mathbf{x} - \mathbf{v}_{p_0})$ to get the Barycentric coordinates. However, a more numerically stable way to solve this problem is by using an LU decomposition [37]. This result can then be used to efficiently solve this system of equations. The matrix A is decomposed as:

$$A = LU. \tag{5-2}$$

However, the LU decomposition is not guaranteed to exist for arbitrary matrices. Therefore a permutation matrix is added to guarantee a solution [38]. This is called the LUP decomposition and is defined as:

$$PA = LU. \tag{5-3}$$

By pre-computing this decomposition for $\mathbf{A_t}$ for each simplex and storing it we can quickly compute the barycentric coordinate transformation for arbitrary simplices.

Solving the system $Ax = b$ turns into solving $LUX = Pb$. Solving this is done in two steps:

1. solve $Ly = Pb$ for y,
2. solve $Ux = y$ for x.

Because L and U are triangular, these equations can be solved by substitution, and no Gaussian elimination is needed. This makes solving $Ax = b$ trivial when the LUP decomposition is known.

5-2-2 Simplex Selection

One of the most time consuming tasks of the model evaluation is the selection of the right simplex for a given Cartesian coordinate. A naive way of finding the right simplex is by calculating the barycentric coordinate for each simplex, and then checking if all of the coordinates are positive (i.e. the coordinate lies within the simplex).

Even though the coordinate transformation was optimized in the previous section, it is still relatively slow. Therefore, another optimization is implemented. The bounding box of each simplex is precomputed by storing the minimum and maximum coordinate values for each dimension. Then, before doing the barycentric coordinate transform, we must verify that the point lies within the bounding box, otherwise the coordinate transform can be skipped.

It can also be noted that, in our fitter, the evaluation of a coordinate is often very close to the previous coordinate. Therefore, there is a big chance the new coordinate lies in the same simplex. This is the reason why the simplex of the previous evaluation is cached and checked first.

5-2-3 ROI Size

When fitting a B-Spline model om some experimental data we have to be careful about the size of the ROI. A Gaussian model can be evaluated for all x and y positions. However, the B-Spline model has bounds to where the model is defined. This means the ROI can be larger than the model, in this case some pixels have to be ignored in the MLE.

Skipping pixels does cause some problems. First, we are not using some information present in the image. This will lead to a loss in precision. Secondly, we have to make sure we the amount of pixels that is used in the MLE stays the same. When fitting a model on experimental data, the model bounds can lie on arbitrary subpixel values. When the estimated x and y positions in the MLE change, this can change which pixels lie inside the model bounds. If this number is not constant, there is a risk of oscillations around the point where this change occurs. To make sure this not happen, models fitted on experimental data are cut off to the nearest whole pixel boundary.

5-3 Initial estimate

Having a good initial estimate for the parameters is important to get fast and reliable fits. A good initial estimate will require less iterations to converge. An initial estimate that is too far from the real parameters can cause the optimization algorithm to get stuck in a local minimum since the optimization problem is not convex. Therefore, a method to get reliable initial estimates for position, intensity and background are required.

First the background is calculated as the lowest intensity in the whole image n:

$$\hat{\text{bg}} = \min(n_1, n_2, \ldots, n_K), \tag{5-4}$$

where K is the total number of pixels. With the estimate for the background known, the intensity is estimated using:

$$\hat{I} = \sum_{i=1}^{K} n_i - K \cdot \hat{\text{bg}}. \tag{5-5}$$

For the estimates of x and y the center off mass is used:

$$\hat{x} = \frac{1}{\sum n_i} \sum_{i=1}^{K} n_i x_i \tag{5-6}$$

where x_i is the x-coordinate of pixel i. y can be computed in a similar fashion. z can be initialized at zero if there are no local minimums in the likelihood function with varying z. Otherwise multiple starting points for z have to be used. To improve robustness we run each fit three times, with z initialized at beginning of the z-range, in the middle and at the end of the z-range.

5-4 Fitting Speed

Besides accuracy, the fitter speed is also an important aspect of the model performance. To compare the different models the speed for Region of Interest (ROI) sizes between 10 and 20 were used. Using each model, a series of images was generated with a emitter position uniformly distributed within 0.5 pixel from the center of the ROI. For the 3D models a z-position of 0.4 µm was used, the middle of the usable z-range. It was found that 10 batches of 10.000 images was enough to show the maximum throughput[1]. The batches need to be large

[1]Hardware used for testing was a Nvidia GTX 1070 GPU, Intel i7 8700k CPU and 32GB of RAM

enough to utilize all CUDA cores in the GPU, and make sure the time spent copying data from the CPU to the GPU is negligible in comparison to the computation time. Smaller batch sizes or a smaller number of batches increased the overhead and limited fitter performance.

The performance of the fitters are shown in figure 5-1. C-Splines are about an order of magnitude slower than the Guassians, the B-Splines are again an order of magnitude slower. Note that the B-Spline performance is still faster than real-time, so this should not pose a bottleneck in practice. Currently the largest bottleneck in the algorithm is the simplex selection. If this can be done instantly (e.g. by using a lookup table), performance should be much closer to the C-Splines.

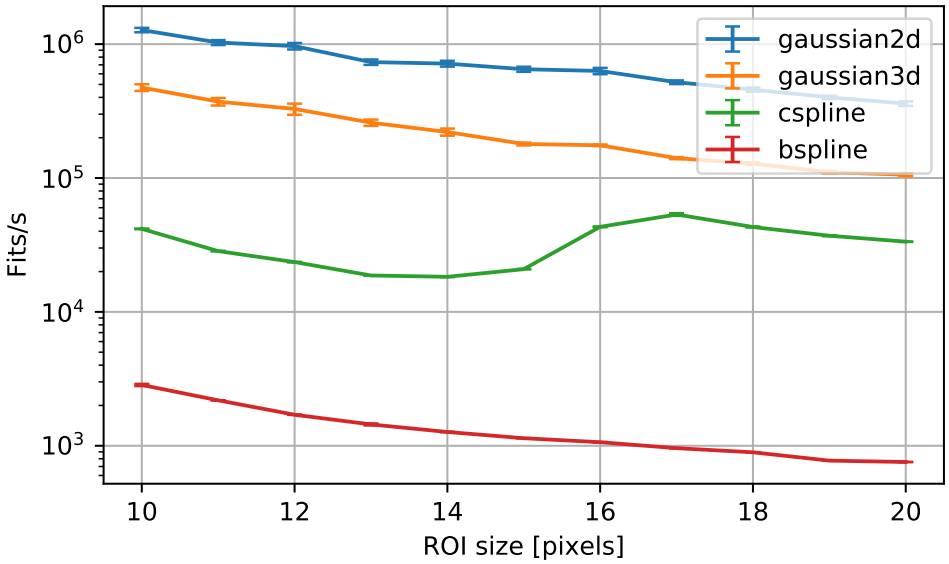


Figure 5-1: Comparison in fitter performance. C-Splines are about an order of magnitude slower than the Guassians, the B-Splines are again an order of magnitude slower. Note that the C-Spline performance increases again at an ROI size of 16, probably due to improved data alignment.

5-5 Numerical Accuracy

The CUDA implementation uses single precision floating point math, which could cause a loss in accuracy. We would like to know what the impact of this on the model quality. Figure 5-2 shows a χ^2 plot, which shows no statistical difference between the two implementations.

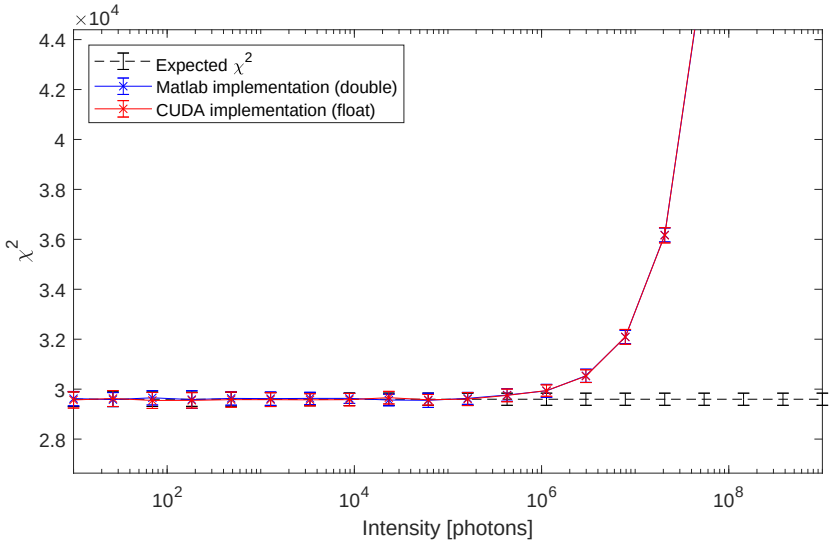


Figure 5-2: Plot of χ^2 values for the Matlab and CUDA implementations.

5-6 Localization Precision

To verify that the fitter implementation works, we can generate images using the vectorial model, and then fit them using our implementation. For 8 different intensities between 10^1 and 10^6, 1000 (x, y) pairs were generated within 1 pixels of the center of the ROI. For all combinations of (x, y) pairs and intensities, three images were generated at $z = -250\,\text{nm}$, $z = 0\,\text{nm}$ and $z = 250\,\text{nm}$. The background intensity was chosen as 1.This results in a total of 24 000 images to fit using the fitter.

After the fitting is done, the estimated locations can be compared to the ground-truth to calculate the standard deviation. We can compute the CRLB according to the B-Spline model at these location to compare to the standard deviation. We call this the experimental CRLB. We can also use the Vectorial model that was used to generate the input data to compute the CRLB, this is also shown in the plots.

Figures 5-3 and 5-5 show the results of these simulation. These plots show that the fitter reaches theoretical uncertainty above an intensity of 1×10^2 photons

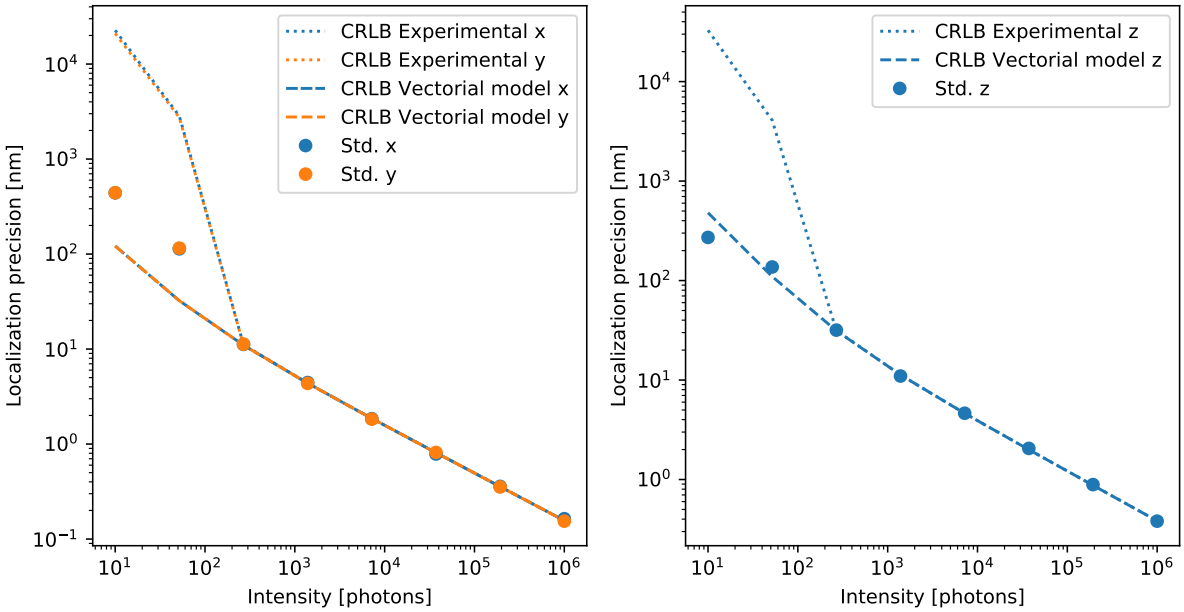


Figure 5-3: This plot shows the fitter reaches theoretical uncertainty. $z = 0\,\text{nm}$

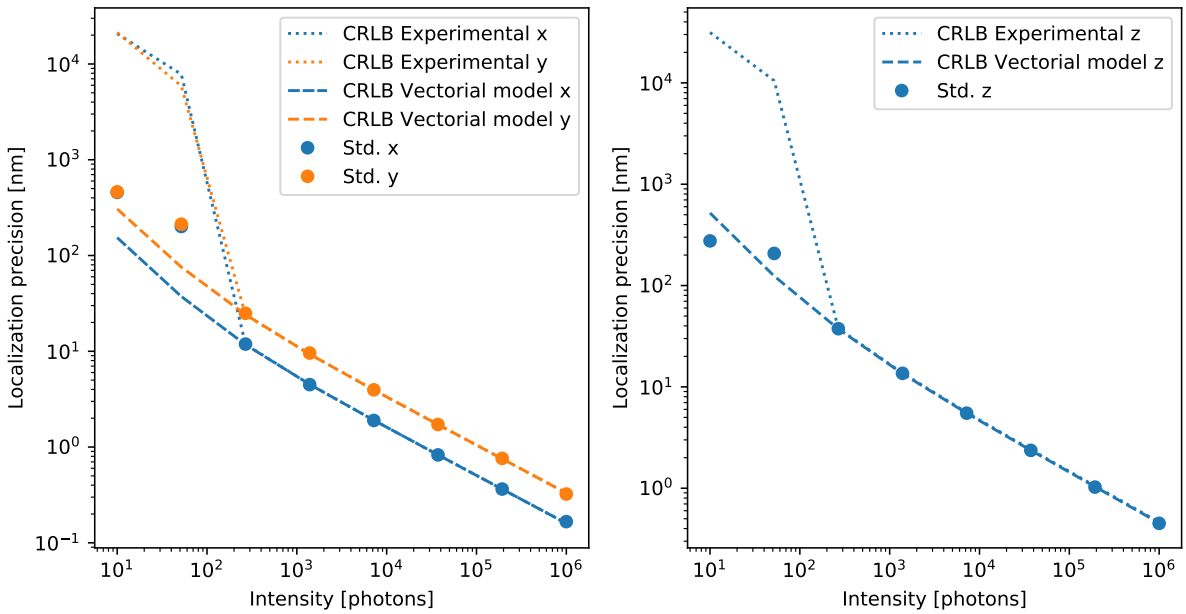


Figure 5-4: This plot shows the fitter reaches theoretical uncertainty. $z = -250\,\mathrm{nm}$

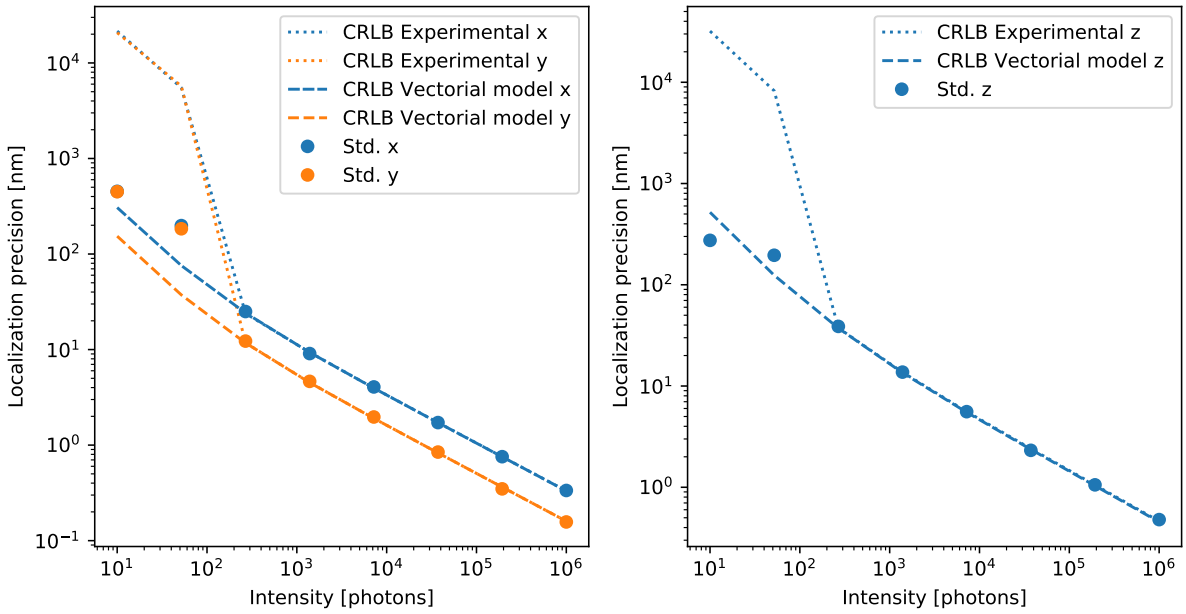


Figure 5-5: This plot shows the fitter reaches theoretical uncertainty. $z = 250\,\mathrm{nm}$

Chapter 6

Experimental Data

6-1 Introduction

This chapter, will look at B-Spline model fitting to experimental data. Even though we tried to match reality in our simulation, there are some things to take into account when using experimental data to create models. There will also be effects we did not anticipate, therefore testing the models on experimental data is essential to proving our method works.

First, we have to convert the data from our camera into actual photon counts. In experimental data we don't know where the beads our located in our data, so we need an automated way to extract the Region of Interest (ROI) containing a bead. Also we don't know the sub-pixel location of each bead, so we need to find the shifts between the different beads we extracted from the z-stack.

The Vectorial model is used to estimate the aberrations present in the microscope. Using the Vectorial model we will generate calibration data, and simulate an experiment we will later do in real life. We can then prove the method works for aberrations present in the microscope, and that our experiment works.

After the procedure is verified on simulated data, we will fit the B-Spline model on actual experimental data. We will use an experiment to prove we reach the Cramér-Rao Lower Bound (CRLB) using our fitter. The localization precision will also be compared to the C-Spline model.

The experimental data used in this chapter was graciously provided by the Ries Group from the European Molecular Biology Laboratory (EMBL).

6-2 Converting Images to Usable Data

In this section, we will describe how to convert images coming from the camera into data we can use for model fitting. This means that the values from the camera have to be converted into the

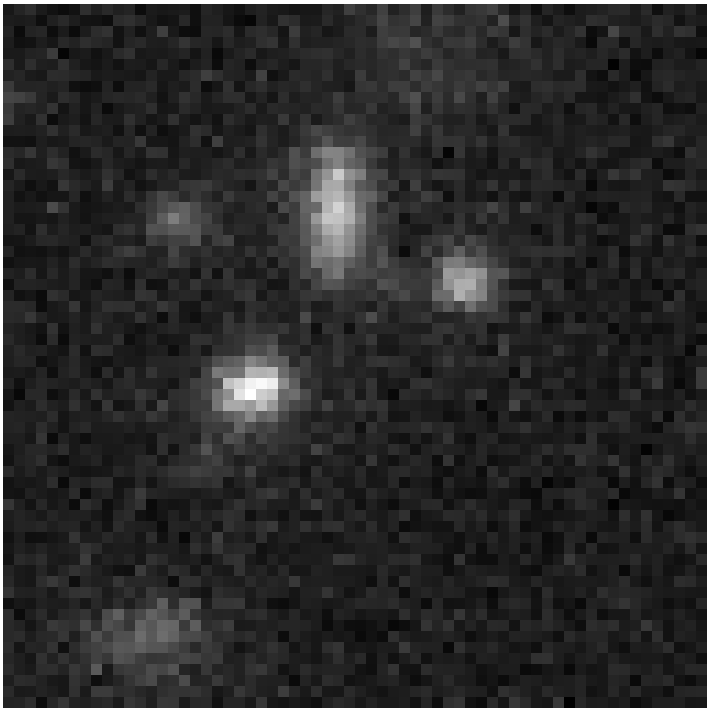


Figure 6-1: A frame from an experimental dataset.

number of photons per pixel. Then, the interesting parts of the Point Spread Function (PSF) have to be extracted from these images. Multiple beads need to be aligned with respect to each other to find their relative translation.

6-2-1 Camera Output to Photons

The first step is to convert the output from the camera into actual photons. The output from the camera is in Analog to Digital Units (ADU). Often, an offset is applied to the camera to make sure low photon counts are not clipped at zero. The first step is to subtract this offset from the input.

The second step is to find the conversion factor for the specific camera. This conversion factor has a unit of photons per ADU (e^-/ADU). This conversion factor includes the quantum efficiency of the camera, and also the gain if present. Optionally, an EM gain may also be present. The number of photons can be calculated using the following formula:

$$\text{number of photons} = (\text{ADU} - \text{offset}) \cdot \frac{\text{conversion factor}}{\text{EM Gain}}. \tag{6-1}$$

6-2-2 Peak Detection

The next step is isolating the beads. This is done using a peak finding algorithm. These regions of interest are then cut from the image. The algorithm is adapted from [39]. The first filtering step consists of:

$$A_1 = \text{uniform}[I, (2\sigma_{\text{PSF}} + 1)] - \text{uniform}[I, 2 \times (2\sigma_{\text{PSF}} + 1)], \tag{6-2}$$

where I is the input image, $\text{uniform}[I, q]$ represents a uniform filter process with a square kernel size q operating on I and σ_{PSF} a estimate of the width of the PSF. A very rough estimate of σ_{PSF} is fine for this algorithm.

The seconds step consists of:

$$A_2 = \max[A_1, 3\sigma_{PSF} + 1], \tag{6-3}$$

where $\max[I, q]$ represents a maximum filter to obtain local maximum values within a square kernel size q.

The center pixel of each region interest found is then found by finding the pixels for which $A_1 = A_2$ and A_2 is larger than some small value to filter out false positives due to noise. Figure 6-2 shows each of the steps in the peak finding process.

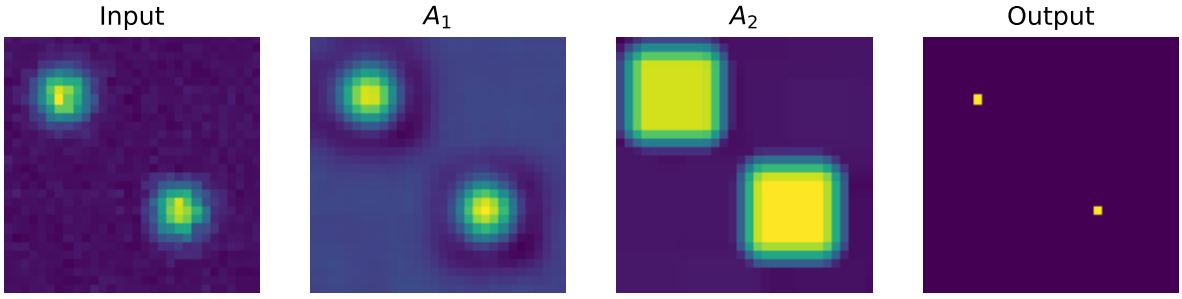


Figure 6-2: The steps in the peak finding algorithm. A_1 shows the subtraction between the uniform filters. A_2 shows the result of the local maximum operation. The output shows the output where $A_1 = A_2$ and is larger than some small value.

6-2-3 Normalize Stacks

Each of the beads in the input stack may have a different intensity, due to varying brightness of the beads. Therefore, the stacks need to be normalized. It is assumed that the central slice is in focus. Each stack is then normalized such that the sum of this central slice is 1.

However, this method does not account for photo bleaching of the beads. During the z-stack the bead will bleach and emit less photons. If we assume the time between two slices in a z-stack is constant we can define the photo bleaching rate [40]:

$$r_{\text{pb}} = ae^{-bz} + ce^{-dz}, \tag{6-4}$$

where z is the number of the slice in the z-stack, a, b, c and d are fitting parameters. The PSF intensity can then be found by dividing the measured intensity by this bleaching rate. Figure 6-3 shows the experimental results of this bleaching on the measured intensity in a z-stack.

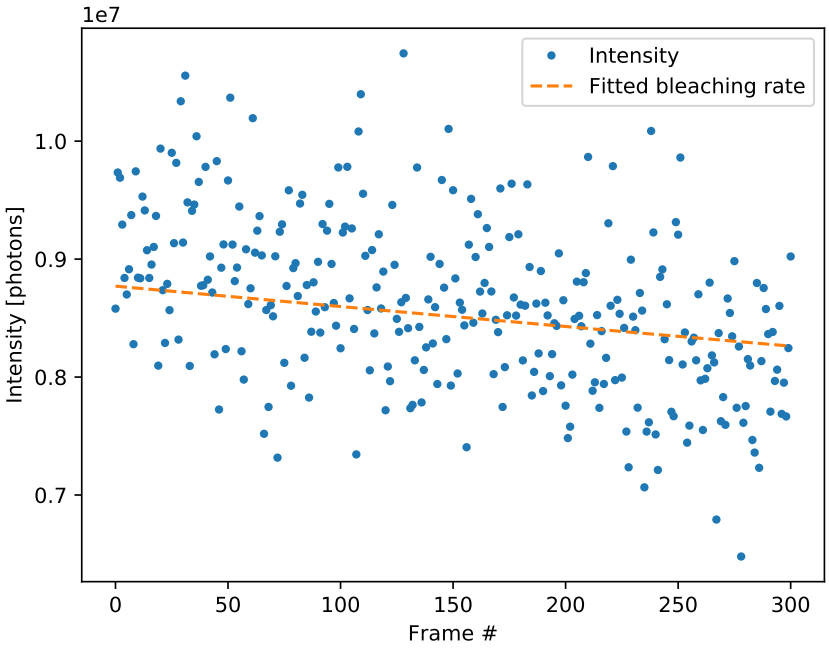


Figure 6-3: Plot showing the decreasing intensity due to photo bleaching in a z-stack.

6-2-4 Find Shifts between Stacks

The extracted beads can still contain small shifts in x and y that need to be identified. Also, the beads might not be attached to the coverslip, so small shifts in z may also be present. The shifts are estimated using an iterative procedure [3].

First, the average of all the beads is calculated. Each bead is aligned to this average using an optimization algorithm (Newton-CG) by maximizing the sum of the product of the two images. The images are shifted in the Fourier domain.

When all the beads are aligned to the average, a new average is calculated from all the translated images. This new average is then used for another round of alignment. This procedure is repeated until all shifts are below 10^{-3} pixel, or after 10 iterations, whichever comes first.

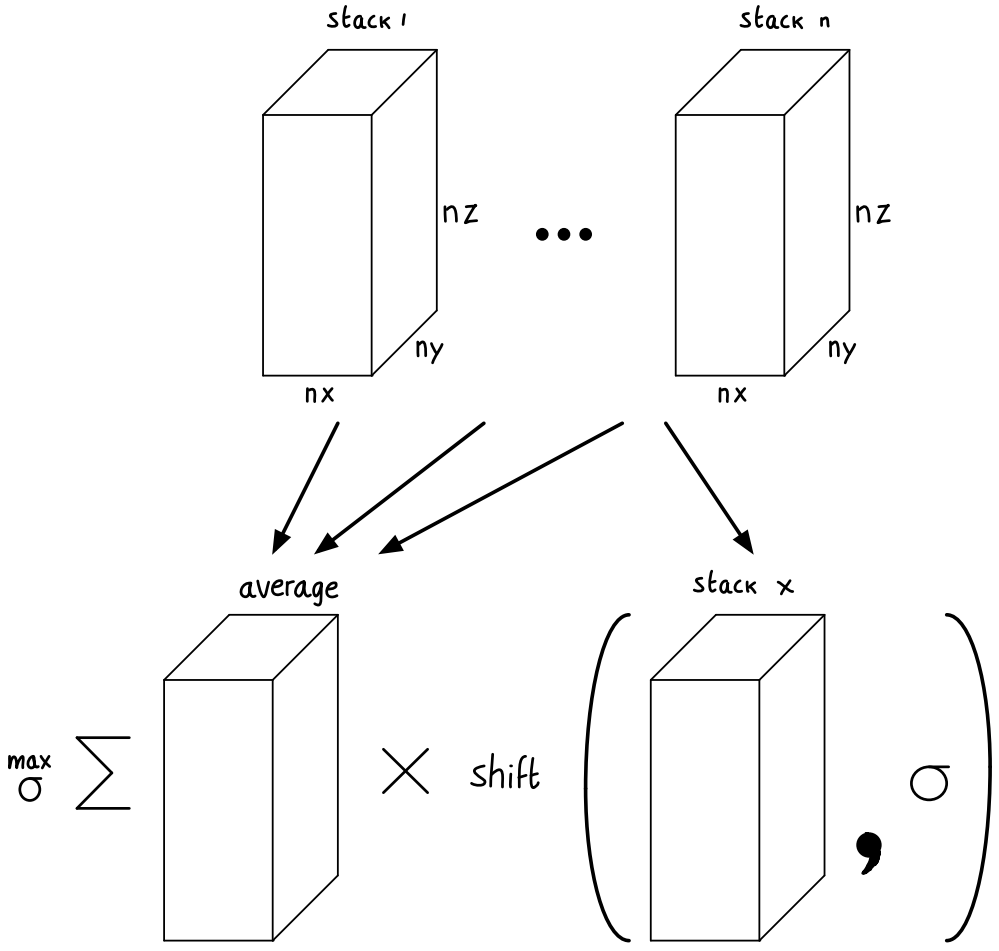


Figure 6-4: Diagram showing the procedure to find the shift for a given stack x. First, all the stacks are averaged, then the shift is found that maximizes the sum of the product of all the pixels. The shift is found for each stack, and a new average is computed. This procedure is repeated until no shifts are needed anymore.

6-2-5 Fit Spline

When the shift of each of the beads is known, the exact (x, y, z) coordinate of each pixel in the input can be calculated. These coordinates, along with their intensity, are then used to fit the spline. In chapter 4 we used a ROI size of 14×14 pixels. However, the input images used in the C-Spline are 19×19. Therefore the B-Spline ROI size was increased to 19×19 to make sure the PSF is also properly modeled for larger z values.

6-3 C-Splines Modeling on Experimental Data

The procedure for the C-Splines is very similar but has a few important differences:

1. The C-Splines do not compensate for bleaching. This won't affect localization, but may impact the estimate of the emitter intensity;

2. After the beads are extracted, they are aligned using a cross correlation. To improve alignment resolution the output of the cross correlation is upsampled 20x by using C-Spline interpolation. The performance of the two shift finding algorithms will be compared in section 6-4;

3. After the shifts have been found, the stacks are shifted by using C-Spline interpolation instead of in the Fourier domain.

4. The stacks are averaged and regularized using smoothing in the z-direction using B-Splines. The average stack is upsampled by a factor of 3. The C-Spline is fitted on this upsampled average stack.

6-4 Comparison of Shift Finding

Our method uses an iterative method of finding the shifts. This method has no discrete values for the shifts like the cross-correlation method, which is limited by the resolution of the correlation and the amount of up-sampling. To compare both methods 32 z-stacks were generated using the Vectorial model. The intensity was chosen at 1×10^5 photons and a background of 20. Each of these stacks has a uniform shift in x and y between 0 and 2.5 pixels, and shift in z between 0 and 5 slices. The magnitude of the error between the estimated and the real shift in (x, y) and in z are compared separately.

The results can be seen in figure 6-5. The iterative method had about 3x lower error than the cross-correlation method in (x, y). However, both methods have errors lower than 1 nm. Both methods severely underestimate the shift in z. Errors in the shift finding will lead to a blurred PSF, which in turn will lead to biased results. Therefore, it is important to keep errors in this stage to a minimum.

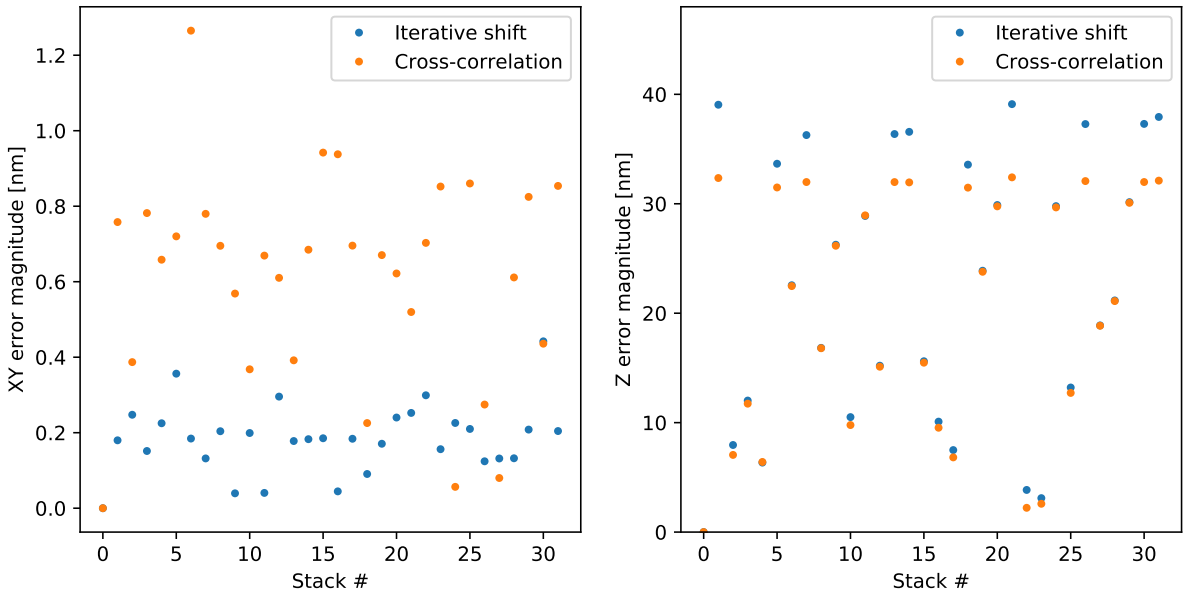


Figure 6-5: Comparison of the two shift finding algorithms. The iterative method from [3] is compared with the cross-correlation and up-sampling method from[4]. Both methods underestimate the shift in z, but the iterative method outperforms the cross-correlation method in x and y

6-5 Simulations Using the Vectorial Model

To verify that the whole process works, we also fitted the same experimental data using the Vectorial model (with both phase and amplitude aberrations). With all the aberrations known, the Vectorial model can also be used to generated new input data to fit a B-Spline model. This was done to verify that all the B-Spline modeling and fitting procedure also works for the specific aberrations present in a real microscope, and not only for the pure astigmatic aberration used in chapter 4.

The same procedure as in chapter 4 was followed to generate reference data (9 z-stacks with combinations of 0, 1/3 and 2/3 of shift in x and y) and fit the spline.

We would like to verify that the standard deviation of the position estimates from the B-Splines fitter reaches the CRLB. Previously we could assume that the ground truth of the emitter locations were known. We could use this to compute the error between the fitted locations and the ground truth, and compute the standard deviation.

In an experimental setup, we don't know the ground truth of our emitter location. Therefore we decided to verify the z-position.

To verify the fitter reaches theoretical uncertainty in z a staircase is used. This dataset consists of a staircase with 40 steps in z, with 100 frames per step and steps of 50 nm. Because there are multiple frames per step in z we can compute the standard deviation of our fits, and compare then with the CRLB. The staircase is shown in figure 6-6.

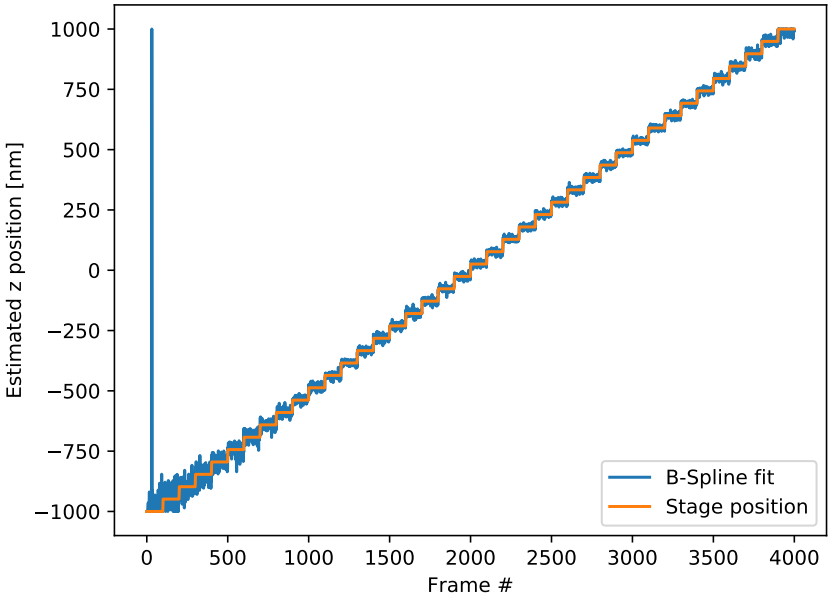


Figure 6-6: Fitted z position versus frame on the simulated staircase.

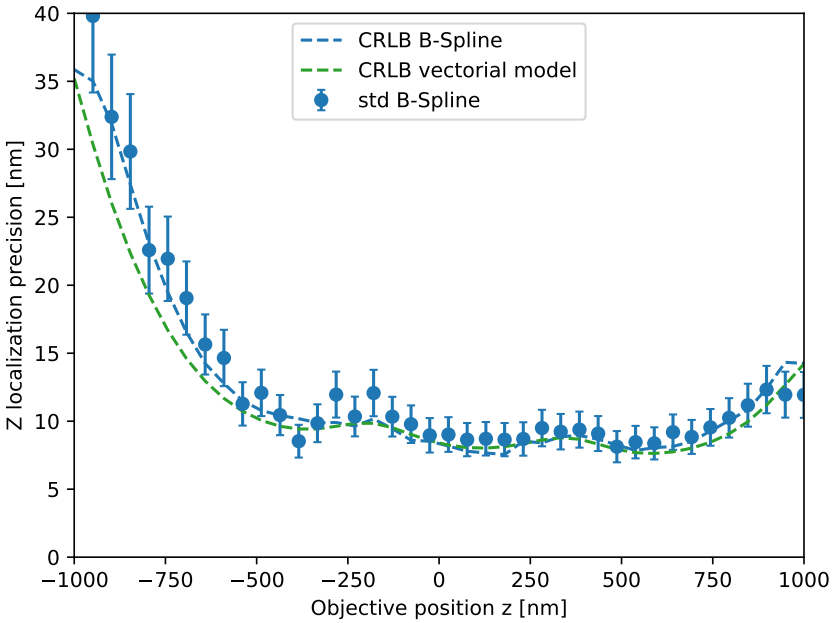


Figure 6-7: CRLB of z-position of fit on simulated data. Each step in z is imaged 100 times and used to calculate standard deviation.

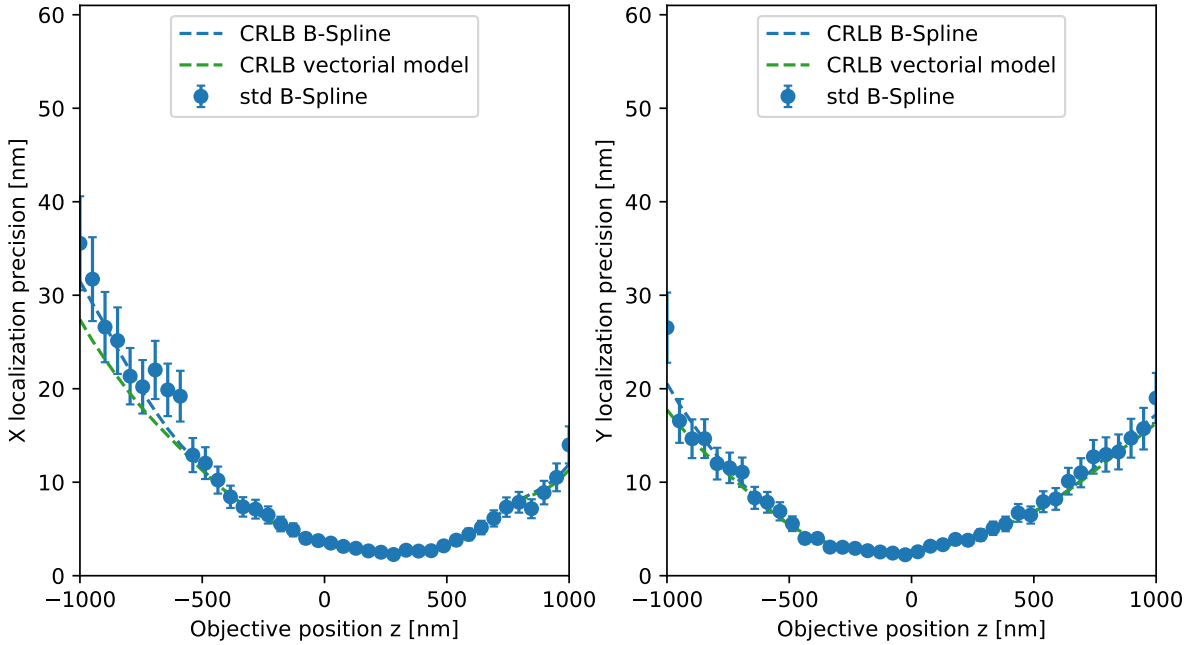


Figure 6-8: CRLB of x and y position of fit on simulated data. Note the standard deviation starts to increase above the CRLB for z values below $-600\,\mathrm{nm}$.

6-6 CRLB on Experimental Data

We would like to know what level of localization precision we can achieve using B-Splines, and compare it to C-Splines. To test this, another set of experimental data is needed. The staircase from the previous section was also acquired on a real microscope.

Figure 6-9 shows the staircase used, and the fitted staircase based on the fitted z-positions. To guarantee that no transients due to stage movements will be present in the result, the middle 50 frames of each step is used to compute the standard deviation of the fits.

The B-Spline model is calibration up till $\pm$ 1000 nm, and the fitter will clip z-values beyond the model range. Therefore the distribution of z-values is biased at the lower end of the staircase.

The C-Spline fitter has trouble at the higher end of the staircase. The extreme z-values have very similar PSFs, therefore it is easy to get stuck in a local minimum. The B-Spline fitter runs each fit multiple starting locations, one at both ends of the range and one in the middle, and chooses the result with the highest likelihood.

The standard deviation is compared with the CRLB in figure 6-10. Both the B-Spline and C-Spline have standard deviations very close to the theoretical CRLB. However, the CRLB and standard deviation for the B-Splines is slightly lower than for the C-Splines. The standard deviations for both methods are slightly lower than the CRLB due an over estimation of the excess noise due to the EM gain.

The uncertainty in the fit error was taken to be a factor of $\sqrt{2/N}$ times the fit error assuming a Gaussian distribution of errors, where N is the number of measurements used for the data point.

The B-Spline model does show biased z values below -600 nm. This can both be seen in the staircase plot as in the CRLB plot. This could be caused by the increase in ROI to match the experimental data. We tried increasing the number of cubes proportional to the increase in ROI, but this did not improve the results. Probably a whole new investigation into the optimal parameters is necessary.

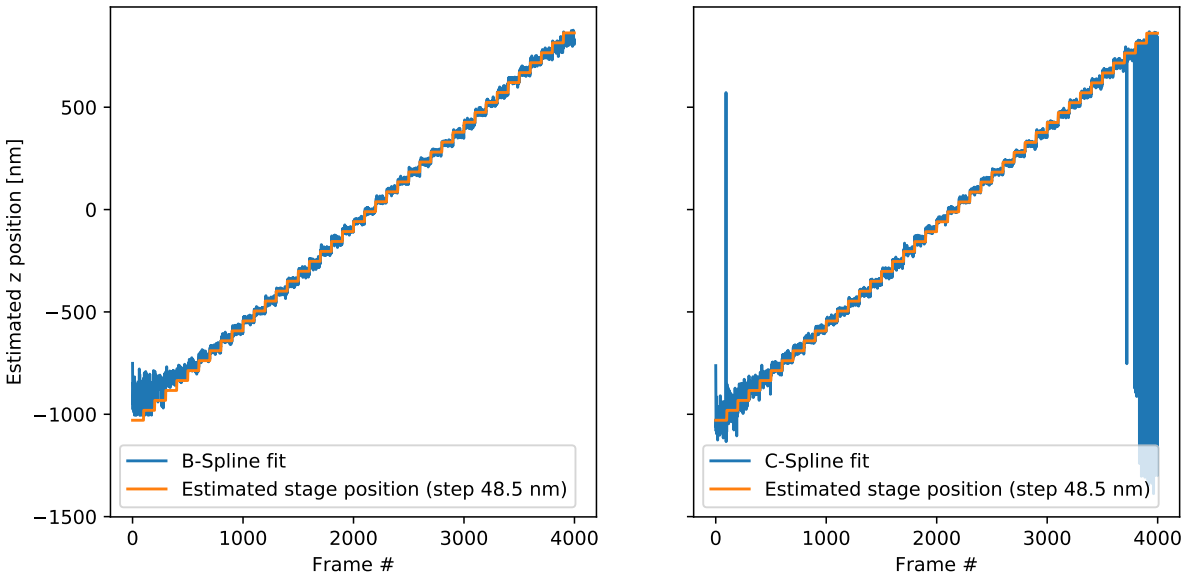


Figure 6-9: Fitted z-position versus frame in the input data for B-Splines and C-Splines. The C-Spline starts to fail for the extreme z-values. Note that the fitted step size is the same for both models.

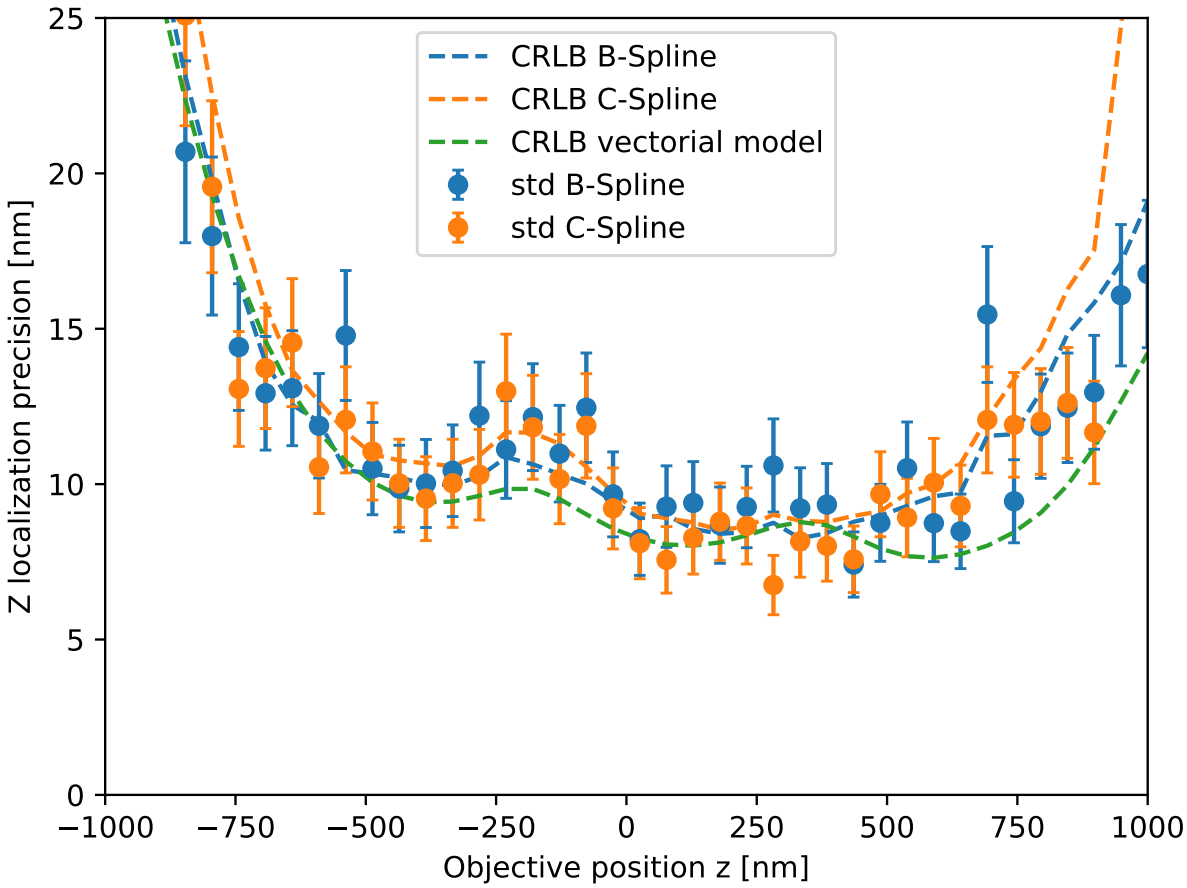


Figure 6-10: CRLB of z-position of fit on experimental data. Each step in z is imaged 100 times and used to calculate standard deviation. This was imaged using an EM gain of 200. The noise from the EM gain was estimated using Poisson noise. Therefore the photon counts need to be divided by the noise excess factor F^2, which approaches 2 for large EM gain [16]. However, this seems to overestimate the noise, therefore both the B-Spline and C-Spline have standard deviations slightly below the CRLB.

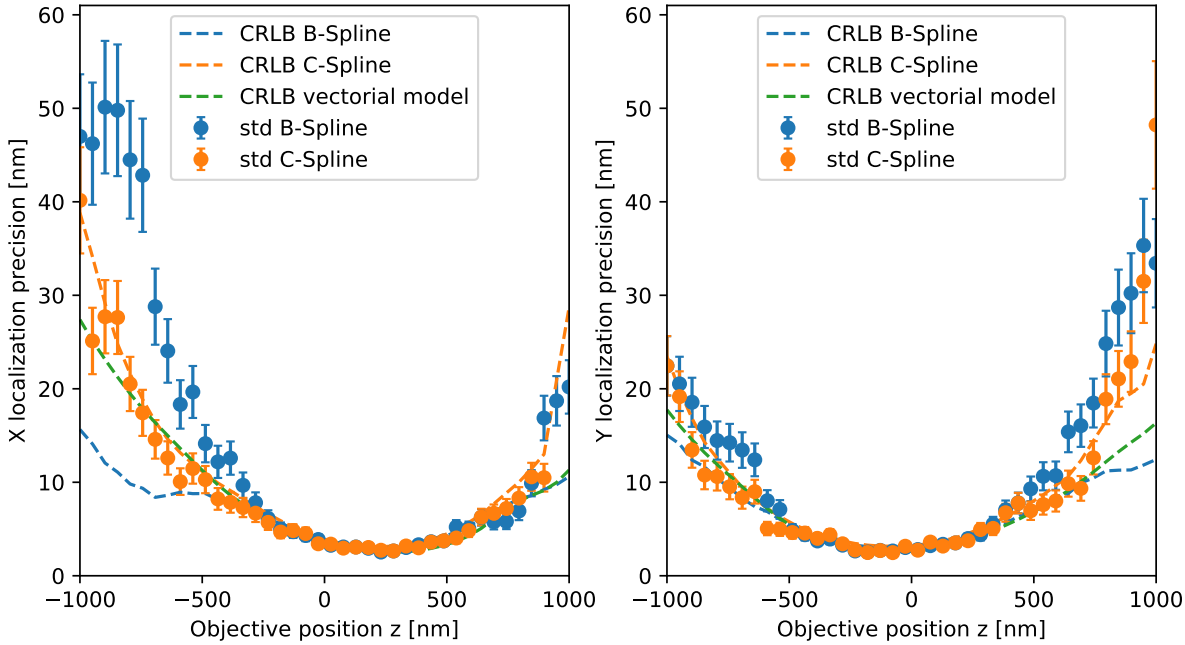


Figure 6-11: CRLB of x and y position of fit on experimental data. Note the standard deviation in x doesn't reach the CRLB below $-600\,\mathrm{nm}$. This was also seen on simulated data. The CRLB is also lower then the CRLB for the vectorial model.

6-7 Chi Square

The χ^2 values can also be computed on experimental data. From the original high-intensity images, lower intensity images can be created using the binomial distribution. Figure 6-12 shows the χ^2 value for different intensities. Both the B-Spline model and C-Spline model are valid about 2×10^3 photons. The B-Spline model performs slightly better than the C-Spline.

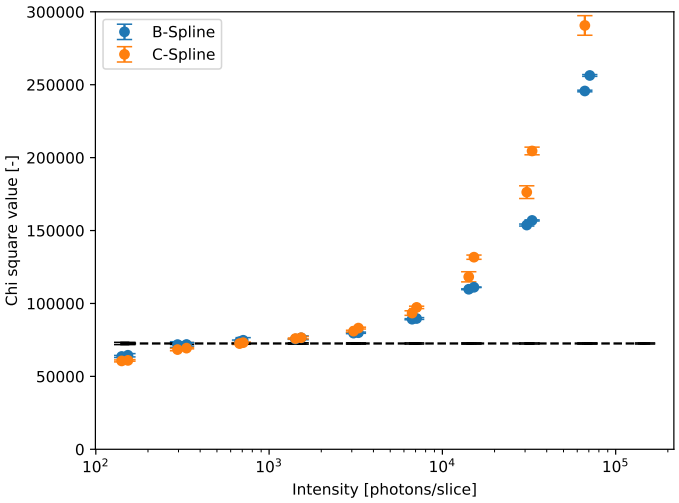


Figure 6-12: χ^2 value for the B-Spline and C-Spline fitted on the experimental bead stacks

6-8 Conclusion

In this chapter, we looked at fitting B-Spline models on experimental data. Using experimental data always poses extra challenges, because some information about the input data has to be estimated (such as the relative position of the beads). In this chapter methods were introduced to pre-process the input data such that all information required for fitting is available.

First, the Vectorial model was used to estimate the aberrations present in the microscope. Using these aberrations a B-Spline model was fitted and images were generated. It was shown that the B-Spline fitter reached theoretical uncertainty.

Then, the B-Spline model was fitted on the experimental data. The standard deviation in z approached the CRLB as reported by the B-Spline model.

The B-Spline models perform slightly better than C-Splines based on localization precision, and uses 144 times[1] fewer spline coefficients.

[1] C-Spline $26 \times 26 \times 300 \times 64 = 12\,979\,200$ coefficients, B-Spline: 90 000 coefficients

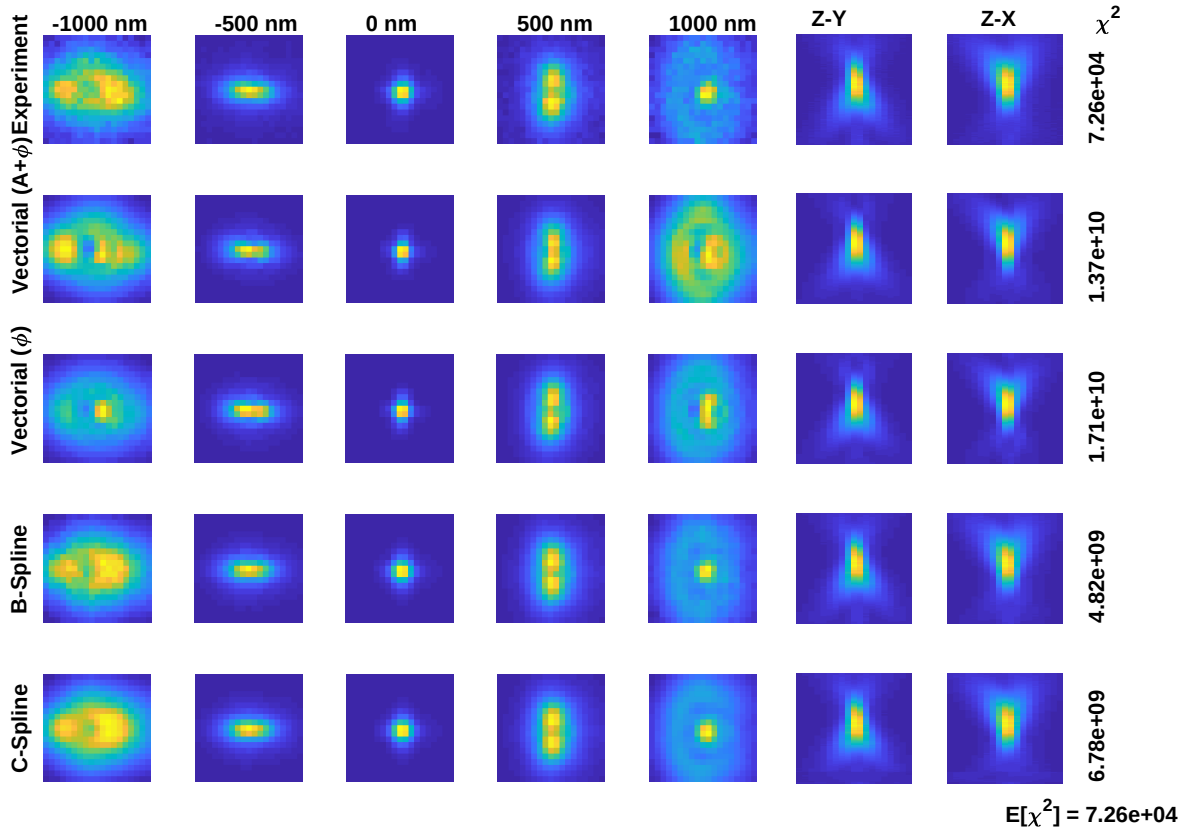


Figure 6-13: Comparison between the experimental data, Vectorial model with both amplitude and phase aberrations, Vectorial model with just phase aberrations ,the B-Spline model and the C-Spline model. The χ^2 for the B-Spline model indicates a significant difference to the experimental data, however the value is much lower than for the Vectorial model. The C-Spline model performs similar to the B-Spline model. The intensity was 1.4×10^5 photons, and a background intensity of 33 photons per pixel.

Chapter 7

Higher Dimensions

7-1 Introduction

In the previous chapters, we assumed the Point Spread Function (PSF) was constant throughout the field of view or in z. However, in practice this is not the case. The PSF varies across the field of view due to changing aberrations [8]. This causes large errors in z, up to 800 nm.

Another source of changes in the PSF is caused by a mismatch in refractive index between the oil immersion objectives used, and the samples mounted in a water based medium. Imaging deeper inside the sample causes spherical aberrations [6]. The magnitude of this spherical aberration is given by:

$$\phi_{\text{aber}} = \frac{2\pi}{\lambda}\left[z_0 n_2 \cos(\theta_2) - z_0 \frac{n_1^2}{n_2}\cos(\theta_1) + z' n_2 \cos(\theta_2)\right], \tag{7-1}$$

where $z_0 = \frac{n_2}{n_1} z_{\text{stage}}$, $z' = z - z_0$, n_1 and n_2 the refractive index of the cover glass and sample medium respectively. See figure 7-1 for the definition of the other variables.

These spherical aberrations need to be corrected for during the fitting, or systematic errors will be introduced. By adding extra parameters to the model, such as position in the field of view or imaging depth we can model these changing aberrations and reduce systematic errors introduced by changes in the PSF.

7-2 Simulation

We would like to estimate the systematic error introduced by these changes in the PSF. To this end, 5 z-stacks were generated. All stacks have an astigmatic aberration of 50 mλ.

However, each of the 5 stacks has increasing spherical aberration (between 0 mλ and 40 mλ). This changing spherical aberration simulates the movement of the focal plane away from the coverslip.

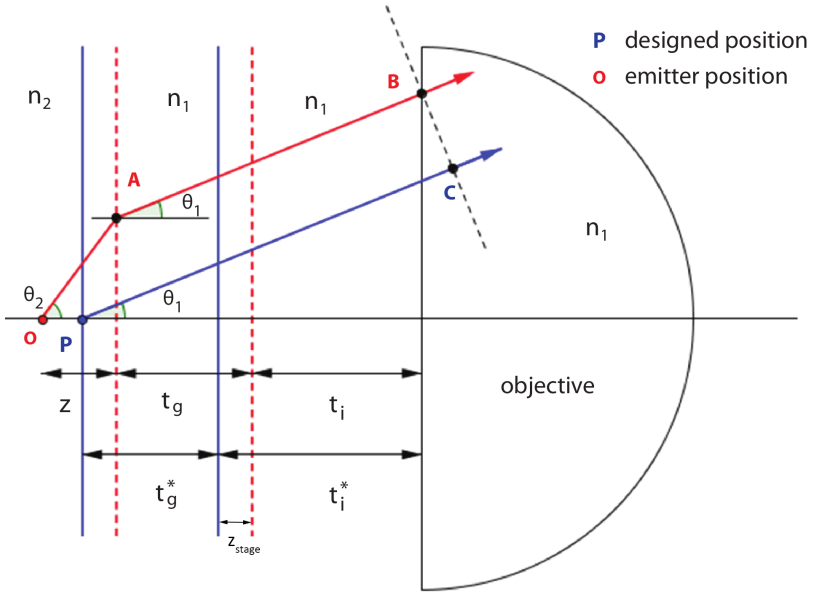


Figure 7-1: Schematic representation of the spherical aberration caused by refractive index mismatch. t_g^* and t_i^* are the designed thickness of the cover glass and immersion medium respectively. The point P is the designed focal point. In order to get the bead in focus the cover glass was moved closer to the objective lens by a distance of $z_{\text{stage}} = t_i^* - t_i$. Image from [6].

For each stack, 100 realizations were generated with an intensity of 1×10^5 and a background of 20. We then used the B-Spline model to fit the positions. The B-Spline model was fitted on z-stacks without spherical aberration, as would be the case in experimental calibration stacks where the beads are attached to the coverslip.

Figure 7-2 shows the results of the 3D B-Spline model, which does not compensate for the spherical aberration. With increasing spherical aberration the Root Mean Square Error (RMSE) in z increases significantly.

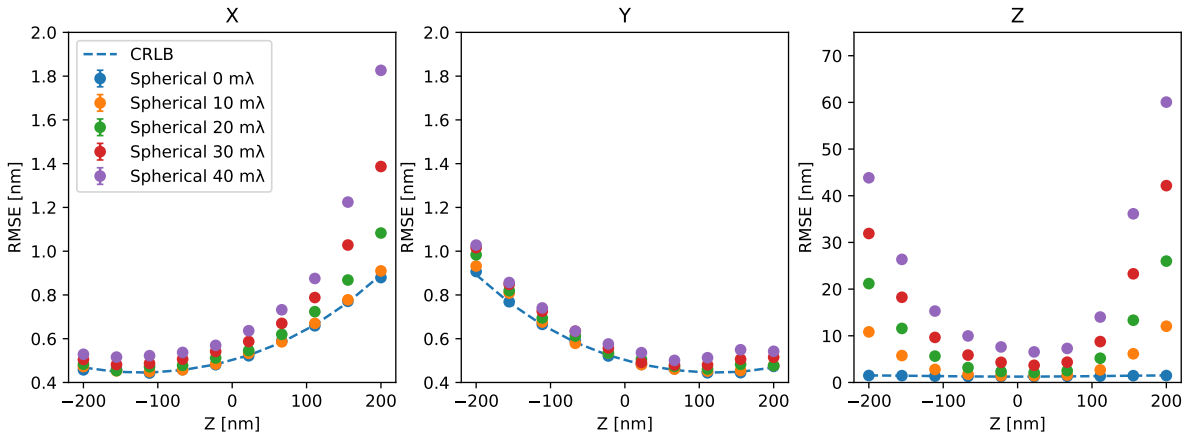


Figure 7-2: Plot of RMSE in x, y and z for changing z-position, and increasing spherical aberration. Note that the RMSE in the z-axis increases significantly with increasing spherical aberration.

7-3 Extending B-Splines to more Dimensions

To remove the bias introduced by the spherical aberrations, we would like to add an extra dimension to the B-Spline model. This dimension can represent the imaging depth in the sample. To generate the input data 10 z-stacks were generated with increasing spherical aberration between 0 mλ and 50 mλ.

The splines were extended with an extra dimension. The number of cubes (M) in x, y and z was kept constant. In the 4^{th} dimension only one cube was used to keep the number of spline coefficients managable. This 4D model has a total of 990 000 coefficients.

7-4 Results from Simulation

The same stacks as in section 7-2 were used. The stage position can be calculated by focusing on the coverslip before the experiment. The change in stage position can then be used as an input to the model. The results are shown in figure 7-3. The 4D model manages to mostly remove the effect of the spherical aberrations. For the 40 mλ spherical aberration, the RMSE is reduced by a factor of 51.

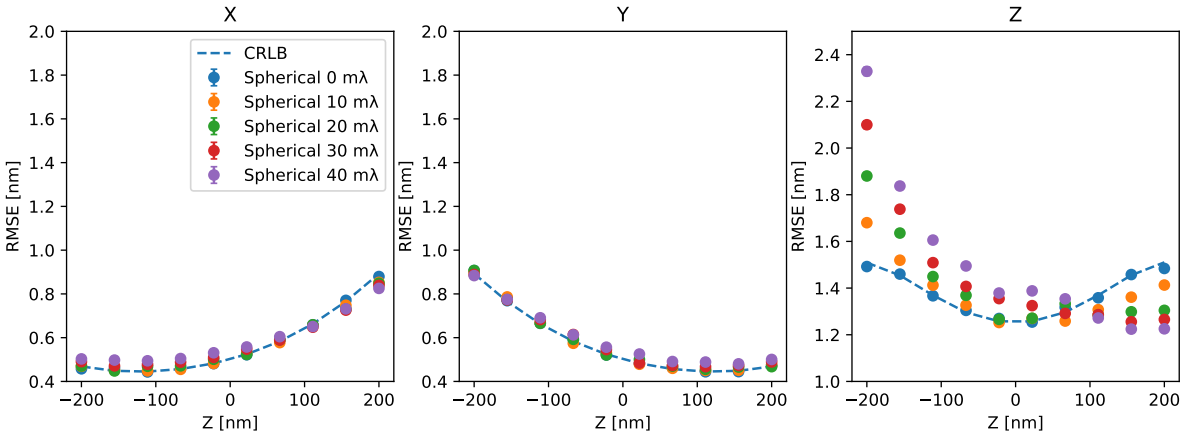


Figure 7-3: Plot of RMSE in x, y and z for changing z position and increasing spherical aberration. Note the difference in scale for the z plot.

7-5 Results from Experimental Data

With the concept proven on simulated data, we would like to show that this also works on experimental data. The experimental data was provided by the Ries Group from the European Molecular Biology Laboratory (EMBL). This is the experimental data used in [7].

The data consists of 64 bead stack at different (x, y) positions. Each stack has 301 steps of 20 nm for a total variation in z of 6 µm. The sample is a piece of agarose gel, with 100 nm beads embedded. The beads are embedded at different z position throughout the gel.

The peak detector from section 6-2-2 is used to find potential beads. Due to the large z-range of the experimental data, the peak finder is ran at 1 µm in the input data.

To estimate the location of the bead a Gaussian fitter was used, with $(x, y, \sigma_x, \sigma_y, I, \text{bg})$ as free parameters. Equation (2-26) was fitted to both σ_x and σ_y. The estimated z location was chosen as the z-slice with the lowest value for z: $\hat{z} = \arg\min \sigma_x(z) + \sigma_y(z)$. If any of the fitters in this step fail to converge, or the estimated z position is too close to the beginning or the end of the z stack the bead is skipped.

Previously we used an alignment procedure using the correlation between the beads. However, this assumes the PSF is constant, which is not the case in this experiment. Therefore, the estimate for the (x, y) location was calculated using the mean of the x and y from the z-slices for which the Gaussian fitter converged successfully, typically within a few hundred nm around the bead. A range of $\pm$ 1 µm around the estimated z-position is extracted

The extracted beads are then used to fit the B-Spline model. Each bead also has an associated stage position, which can be used as the fourth dimension for the model. Both a 3D model and a 4D model are fitted to measure the difference.

From the input data 8 stacks are used to calibrate the model. The other half was use to measure the results. To verify the presence of depth dependent aberrations, the 3D model was used to localize these beads (figure 7-4). We expect the estimated z-positions to match the stage position. However, we note that for beads deeper in the sample the z-position is underestimated with errors up to 200 nm. We also note that the over and underestimation is correlated with the distance of the focal plane above the coverslip.

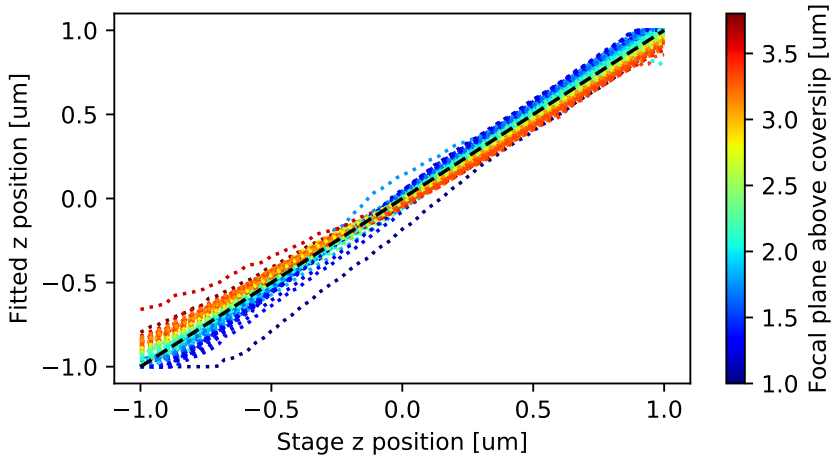


Figure 7-4: Fitted z-position compared to stage z-position. There is a correlation between the distance of the focal plane above the coverslip and the over or under estimation of the z position. This indicates depth dependent aberrations causing errors.

To solve this problem, the beads were fitted using the 4D model (figure 7-6). The 4D model removes the depth-dependency present in the 3D model. Unfortunately the 4D model has a higher variance in the fitted z-position. This is probably due to the lower degree (5 instead of 7) and low resolution of the triangulation of the fourth dimension. However, with the current model we were already hitting computational constraints. More efficient fitting routines are available, but are not yet compatible with continuity orders

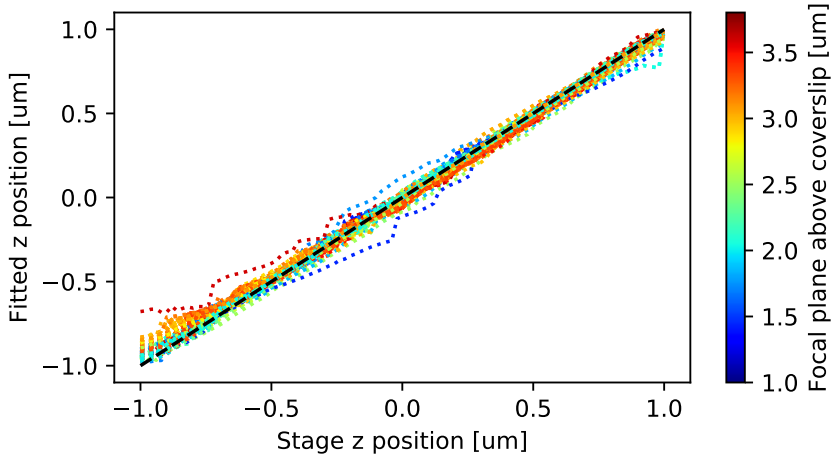


Figure 7-5: Fitted z-position compared to stage z-position. There is a correlation between the distance of the focal plane above the coverslip and the over or under estimation of the z-position. This indicates depth dependent aberrations causing errors.

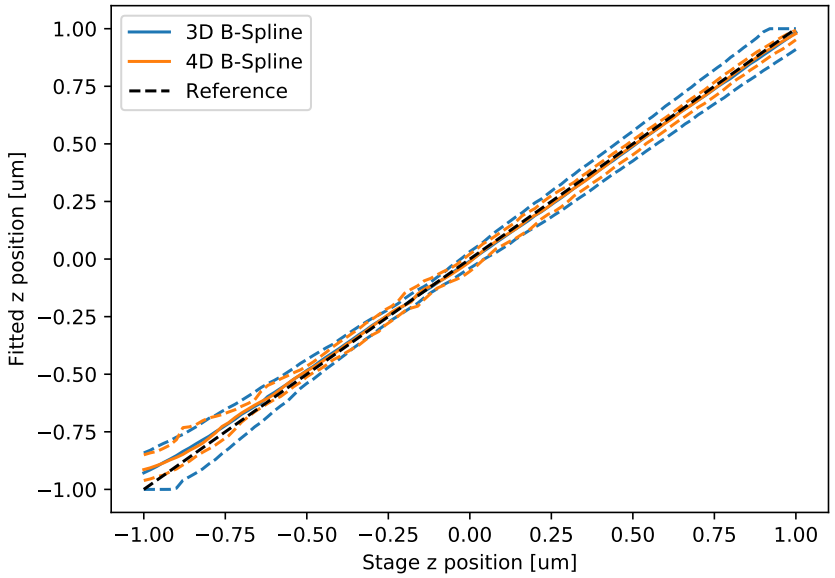


Figure 7-6: Fitted z-position compared to stage z-position. For both the 3D and 4D model, the mean, 5th and 95th percentiles are shown.

7-6 Conclusion

In this chapter, we have shown that B-Splines can be extended with an extra dimension to increase robustness against varying aberration due to imaging depth or variations in the field of view. Adding this to the model significantly removed the effect of these errors.

Unfortunately, due to computational constraints only one extra dimension could be added. But with a more efficient implementation of the B-Spline model fitting routines, more dimensions could be added. For example, this could be used to encode both the x and y position in the field of view and the imaging depth.

Chapter 8

Conclusion

Single Molecule Localization Microscopy (SMLM) is a very powerful technique that allows imaging with a resolution beyond the diffraction limit. One of the key parts of SMLM is having a good model of the Point Spread Function (PSF). The wrong PSF model can reduce precision, or even introduce significant biases in the result. Therefore different models were studied, and their performance evaluated.

A model based on Simplex B-Splines was introduced. B-Splines are a very flexible modeling technique. The number of coefficients is much lower than current state of the art methods, such as C-Splines. B-Splines can also be extended to higher dimensions.

The effect of different tuning parameters, such as spline degree, triangulation and continuity order on the model quality were studied. The models were compared to the reference data using a statistical test (χ^2 test). This statistical test tells up till which image intensity a model is indistinguishable from the reference data.

The B-Splines were implemented in a Maximum Likelihood Estimation (MLE) algorithm. Using simulated data, it was shown that the fitter reaches the theoretical lower bound for the precision, the Cramér-Rao Lower Bound (CRLB). The model evaluation code was written in CUDA to run on a GPU. This allows faster than real-time fitting performance.

To create B-Splines models from experimental data, a procedure was developed to automatically extract beads from experimental data. The beads are normalized and aligned. With the exact location known, the B-Spline coefficients are fitted.

The performance of the B-Spline model was also verified on experimental data. Using the Vectorial model the aberrations present in the microscope used for the experiments was estimated. The experiment was first verified using simulations with these specific aberrations. Then, the actual experimental data was used. The B-Splines performed similar to the state of the art methods (C-Splines), while using 144 times less spline coefficients.

Most applications assume the PSF is constant over the field of view. However when imaging deep inside the sample, a spherical aberration is introduced due to refractive index mismatches. These aberrations cause significant bias in the z estimate if not corrected. A fourth dimension

was added to the B-Splines model to represent the distance of the focal plane above the coverslip. This reduced the errors due to the spherical aberrations by a factor of 51 in simulations.

The effects of depth dependent aberrations were also shown on experimental data. A 4D B-Spline model was created to model these aberrations. This significantly reduced the correlation between the position of the focal plane above the coverslip.

The experiments in this thesis show that the B-Splines perform as well as state of the art methods, while using less coefficients. Their flexibility allows extension to higher dimension, something that is not feasible with current methods.

8-1 Outlook

The current implementation of the B-Spline fitter is about an order of magnitude slower then the C-Spline implementation. By using a smarter way to select the right simplex this performance gap might be closed.

When fitting the 4D B-Spline model, we were hitting the computational limits of the current coefficient fitter implementation. If we want to increase the dimension further a more efficient coefficient fitter has to be used. For example the use of distributed solver as in [41] can be investigated.

Another method to reduce the computational demand is to make use of simplotope splines [42, 43]. Simplex splines allow varying the structure of the model over the domain. In more complex regions, the simplices can be packed more densely to increase the approximation power. However, the spline degree has to be constant over the whole simplex spline.

If it is desired to vary the spline degree over the domain it is possible to make use of simplotope splines. Just as simplex splines are built by combining splines on multiple splines, simplotope splines are built by combining multiple simplex splines. This allows the degree to be varied over the domain. For example the variations of the PSF over the field might be small, and may allow using a lower spline degree.

Another area of future research is an on-line update scheme of the model. The user can then start with a generic model, and the model will adapt to the specific microscope during the experiment. This is especially relevant when the localization algorithm is running real-time (e.g. showing results while the user is doing the experiment).

Instead of using high intensity bead stacks for calibration, we would like to use data from biological experiment. This experimental data has much lower photon counts. When fitting the spline coefficients we assume the residuals are white noise. However, in practice this is not the case due to the Poisson noise from the imaging process. This can be solved by changing the solver for the spline coefficients to take this into account, but this will make solving computationally more expensive. Another solution will be to pre-process the data by transforming the Poisson distributed noise into a Gaussian distributed noise using the Anscombe Transform, this was done successfully for SMLM in [44].

Appendix A

Scripts

A-1 Type I Triangulation

```
#!/usr/bin/env python3
import numpy as np
import math
import scipy.io

D = 4 # Dimension

def add_one(cur):
    """Generates all possible next steps in the path"""
    r = []
    for i in range(len(cur)):
        c = cur[:]
        if c[i] == 0:
            c[i] = 1
            r.append(c)
    return r


def get_int(row):
    """Convert array of 1s and 0s to integer representation"""
    return int(''.join(map(str, row)), 2)

def build_paths(cur):
    """Recursive function to build all paths"""
    paths = []
    for a in add_one(cur):
        built = build_paths(a)
        for b in built:
            paths.append([cur] + b)
    if len(paths) == 0:
```

```
31          paths.append([cur])
32      return paths
33
34  # Build vertex matrix
35  vertices = np.zeros((2**D, D), dtype=np.int)
36  for i in range(2**D):
37      s = bin(i)[2:].rjust(D, '0')
38      vertices[i] = list(map(int, s))
39
40
41  # Fill edges matrix based on paths
42  edges = np.zeros((math.factorial(D), D+1), dtype=np.int)
43  paths = build_paths([0] * D)
44  for i, path in enumerate(paths):
45      for j, v in enumerate(path):
46          idx = get_int(v)
47          edges[i, j] = idx
48
49  print(vertices)
50  print(edges + 1) # +1 to convert to Matlab based indexing
51
52  scipy.io.savemat('4d.mat', {"VERTICES": vertices.astype(np.float), "EDGES
      ": edges.astype(np.float)+1})
```

Bibliography

[1] E. Betzig, G. H. Patterson, R. Sougrat, O. W. Lindwasser, S. Olenych, J. S. Bonifacino, M. W. Davidson, J. Lippincott-Schwartz, and H. F. Hess, "Imaging Intracellular Fluorescent Proteins at Nanometer Resolution," *Science*, vol. 313, pp. 1642–1645, Sept. 2006.

[2] S. Stallinga and B. Rieger, "Accuracy of the Gaussian Point Spread Function model in 2d localization microscopy," *Optics Express*, vol. 18, p. 24461, Nov. 2010.

[3] H. P. Babcock and X. Zhuang, "Analyzing Single Molecule Localization Microscopy Data Using Cubic Splines," *Scientific Reports*, vol. 7, Dec. 2017.

[4] Y. Li, M. Mund, P. Hoess, J. Deschamps, U. Matti, B. Nijmeijer, V. J. Sabinina, J. Ellenberg, I. Schoen, and J. Ries, "Real-time 3d single-molecule localization using experimental point spread functions," *Nature Methods*, vol. 15, pp. 367–369, Apr. 2018.

[5] C. C. De Visser, *Global nonlinear model identification with multivariate splines: application to aerodynamic model identification of the Cessna Citation II.* PhD thesis, [s.n.], S.l., 2011. OCLC: 840446321.

[6] S. Liu, E. B. Kromann, W. D. Krueger, J. Bewersdorf, and K. A. Lidke, "Three dimensional single molecule localization using a phase retrieved pupil function," *Optics Express*, vol. 21, p. 29462, Dec. 2013.

[7] Y. Li, Y.-L. Wu, P. Hoess, M. Mund, and J. Ries, "Depth-dependent PSF calibration and aberration correction for 3d single-molecule localization," *bioRxiv*, Feb. 2019.

[8] T. Yan, C. J. Richardson, M. Zhang, and A. Gahlmann, "Computational Correction of Spatially-Variant Optical Aberrations in 3d Single Molecule Localization Microscopy," *bioRxiv*, Dec. 2018.

[9] M. E. Siemons, "High precision wavefront control for 4d PSF engineering," p. 76.

[10] Wikipedia, "Point spread function — Wikipedia, the free encyclopedia." http://en.wikipedia.org/w/index.php?title=Point%20spread%20function&oldid=884764140, 2019. [Online; accessed 08-April-2019].

[11] F. Fok, "Comparing Point Spread Function Models for Localization Microscopy," p. 64.

[12] S. Vladimir, "Amateur Telescope Optics." https://www.telescope-optics.net/monochromatic_eye_aberrations.htm, July 2006.

[13] S. Ghosh and C. Preza, "Characterization of a three-dimensional double-helix point-spread function for fluorescence microscopy in the presence of spherical aberration," *Journal of Biomedical Optics*, vol. 18, p. 036010, Mar. 2013.

[14] Y. Shechtman, L. E. Weiss, A. S. Backer, S. J. Sahl, and W. E. Moerner, "Precise Three-Dimensional Scan-Free Multiple-Particle Tracking over Large Axial Ranges with Tetrapod Point Spread Functions," *Nano Letters*, vol. 15, pp. 4194–4199, June 2015.

[15] C. Smith, M. Huisman, M. Siemons, D. Grünwald, and S. Stallinga, "Simultaneous measurement of emission color and 3d position of single molecules," *Optics Express*, vol. 24, p. 4996, Mar. 2016.

[16] S. Fullerton, K. Bennett, E. Toda, and T. Takahashi, "Optimization of precision localization microscopy using CMOS camera technology," (San Francisco, California, USA), p. 82280T, Feb. 2012.

[17] S. Shashkova and M. Leake, "Single-molecule fluorescence microscopy review: shedding new light on old problems," *Bioscience Reports*, vol. 37, p. BSR20170031, Aug. 2017.

[18] C. P. Toseland, "Fluorescent labeling and modification of proteins," *Journal of Chemical Biology*, vol. 6, pp. 85–95, July 2013.

[19] C. Cremer and B. R. Masters, "Resolution enhancement techniques in microscopy," *The European Physical Journal H*, vol. 38, pp. 281–344, Apr. 2013.

[20] S. T. Hess, T. P. Girirajan, and M. D. Mason, "Ultra-High Resolution Imaging by Fluorescence Photoactivation Localization Microscopy," *Biophysical Journal*, vol. 91, pp. 4258–4272, Dec. 2006.

[21] M. J. Rust, M. Bates, and X. Zhuang, "Sub-diffraction-limit imaging by stochastic optical reconstruction microscopy (STORM)," *Nature Methods*, vol. 3, pp. 793–796, Oct. 2006.

[22] G. T. Dempsey, M. Bates, W. E. Kowtoniuk, D. R. Liu, R. Y. Tsien, and X. Zhuang, "Photoswitching Mechanism of Cyanine Dyes," *Journal of the American Chemical Society*, vol. 131, pp. 18192–18193, Dec. 2009.

[23] J. W. Goodman, *Introduction to Fourier Optics*. McGraw-Hill, 2 ed., Sept. 1996.

[24] B. M. Hanser, M. G. L. Gustafsson, D. A. Agard, and J. W. Sedat, "Phase-retrieved pupil functions in wide-field fluorescence microscopy," *Journal of Microscopy*, vol. 216, pp. 32–48, Oct. 2004.

[25] T. A. Laurence and B. A. Chromy, "Efficient maximum likelihood estimator fitting of histograms," *Nature Methods*, vol. 7, pp. 338–339, May 2010.

[26] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly of Applied Mathematics*, vol. 2, pp. 164–168, July 1944.

[27] D. W. Marquardt, "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," *Journal of the Society for Industrial and Applied Mathematics*, vol. 11, pp. 431–441, June 1963.

[28] Y. Shechtman, S. J. Sahl, A. S. Backer, and W. Moerner, "Optimal Point Spread Function Design for 3d Imaging," *Physical Review Letters*, vol. 113, Sept. 2014.

[29] R. J. Noll, "Zernike polynomials and atmospheric turbulence," *Journal of the Optical Society of America*, vol. 66, p. 207, Mar. 1976.

[30] S. R. P. Pavani, M. A. Thompson, J. S. Biteen, S. J. Lord, N. Liu, R. J. Twieg, R. Piestun, and W. E. Moerner, "Three-dimensional, single-molecule fluorescence imaging beyond the diffraction limit by using a double-helix point spread function," *Proceedings of the National Academy of Sciences*, vol. 106, pp. 2995–2999, Mar. 2009.

[31] C. Smith, *Optimum Single Molecule Localization Microscopy*. PhD thesis.

[32] C. S. Smith, N. Joseph, B. Rieger, and K. A. Lidke, "Fast, single-molecule localization that achieves theoretically minimum uncertainty," *Nature Methods*, vol. 7, pp. 373–375, May 2010.

[33] J. R. Shewchuk, "General-Dimensional Constrained Delaunay and Constrained Regular Triangulations, I: Combinatorial Properties," *Discrete & Computational Geometry*, vol. 39, pp. 580–637, Mar. 2008.

[34] M. d. Berg, ed., *Computational geometry: algorithms and applications*. Berlin: Springer, 3rd ed ed., 2008.

[35] P. Scott Mara, "Triangulations for the cube," *Journal of Combinatorial Theory, Series A*, vol. 20, pp. 170–177, Mar. 1976.

[36] N. Yamashita and M. Fukushima, "On the Rate of Convergence of the Levenberg-Marquardt Method," in *Topics in Numerical Analysis* (G. Alefeld and X. Chen, eds.), vol. 15, pp. 239–249, Vienna: Springer Vienna, 2001.

[37] A. Schwarzenberg-Czerny, "On matrix factorization and efficient least squares solution.," *Astronomy and Astrophysics Supplement*, p. 405, Apr. 1995.

[38] P. Okunev and C. R. Johnson, "Necessary And Sufficient Conditions For Existence of the LU Factorization of an Arbitrary Matrix," *arXiv:math/0506382*, June 2005. arXiv: math/0506382.

[39] F. Huang, S. L. Schwartz, J. M. Byars, and K. A. Lidke, "Simultaneous multiple-emitter fitting for single molecule super-resolution imaging," *Biomedical Optics Express*, vol. 2, p. 1377, May 2011.

[40] L. Song, E. Hennink, I. Young, and H. Tanke, "Photobleaching kinetics of fluorescein in quantitative fluorescence microscopy," *Biophysical Journal*, vol. 68, pp. 2588–2600, June 1995.

[41] C. C. de Visser, E. Brunner, and M. Verhaegen, "On distributed wavefront reconstruction for large-scale adaptive optics systems," *Journal of the Optical Society of America A*, vol. 33, p. 817, May 2016.

[42] M. Van Den Aarssen, "Distributed Approach for Aerodynamic Model Identification of the ICE Aircraft," p. 150.

[43] T. Visser, C. C. de Visser, and E.-J. van Kampen, "Towards the multivariate simplotope spline: continuity conditions in a class of mixed simplotopic grids," *arXiv:1602.01335 [math]*, Feb. 2016. arXiv: 1602.01335.

[44] U. Köthe, F. Herrmannsdörfer, I. Kats, and F. A. Hamprecht, "SimpleSTORM: a fast, self-calibrating reconstruction algorithm for localization microscopy," *Histochemistry and Cell Biology*, vol. 141, pp. 613–627, June 2014.

Glossary

List of Acronyms

SMLM	Single Molecule Localization Microscopy
PSF	Point Spread Function
CRLB	Cramér-Rao Lower Bound
NA	Numerical Aperture
OTF	Optical Transfer Function
MLE	Maximum Likelihood Estimation
ROI	Region of Interest
GFP	Green Fluorescent Protein
PALM	PhotoActivated Localization Microscopy
STORM	Stohastic Optical Reconstruction Microscopy
PA-GFP	Photo-Activatable Green Fluorescent Protein
ADU	Analog to Digital Units
RMSE	Root Mean Square Error
DMD	Digital Micromirror Device
SLM	Spatial Light Modulator